

AD \_\_\_\_\_

Award Number: DAMD17-99-C-9001

TITLE: Defense Healthcare Information Assurance Program

PRINCIPAL INVESTIGATOR: Archie Andrews, Lynn Crane, Jack Stinson,  
Steve Pellissier, Steve Packard, Chris  
Alberts, David Fisher, Robert Rosenstein,  
Thornton White, Keith McCall, Pat Wise

CONTRACTING ORGANIZATION: Advanced Technology Institute  
North Charleston, South Carolina 29148

REPORT DATE: November 2000

TYPE OF REPORT: Annual

PREPARED FOR: U.S. Army Medical Research and Materiel Command  
Fort Detrick, Maryland 21702-5012

DISTRIBUTION STATEMENT: Approved for public release;  
Distribution unlimited

The views, opinions and/or findings contained in this report are those of the author(s) and should not be construed as an official Department of the Army position, policy or decision unless so designated by other documentation.

20010223 099

<b>REPORT DOCUMENTATION PAGE</b>			Form Approved OMB No. 0704-0188	
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.				
1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE November 2000	3. REPORT TYPE AND DATES COVERED Annual (15 Oct 99 - 14 Oct 00)		
4. TITLE AND SUBTITLE Defense Healthcare Information Assurance Program (DHIAP)		5. FUNDING NUMBERS DAMD17-99-C-9001		
6. AUTHORS Archie Andrews, Lynn Crane, Jack Stinson, Steve Pellissier, Steve Packard, Chris Alberts, David Fisher, Robert Rosenstein, Thornton White, Keith McCall, Pat Wise				
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Advanced Technology Institute North Charleston, South Carolina 29148		8. PERFORMING ORGANIZATION REPORT NUMBER		
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) U.S. Army Medical Research and Materiel Command Fort Detrick, Maryland 21702-5012		10. SPONSORING/MONITORING AGENCY REPORT NUMBER		
11. SUPPLEMENTARY NOTES				
12a. DISTRIBUTION/AVAILABILITY STATEMENT Approved for Public Release; Distribution Unlimited		12b. DISTRIBUTION CODE		
13. ABSTRACT (Maximum 200 words) This Annual Report covers the activities and accomplishments of the Defense Healthcare Information Assurance Program (DHIAP) for the period 16 October 1999 to 15 October 2000. It describes in detail the completion of DHIAP Phase I tasks (Technical Assessment, Prototype Design, Technology Demonstration, and Technology Transition) and summarizes the DHIAP Phase II work started in April 2000 (Risk Analysis, Business Case Analysis, and Simulation Capability). In Risk Analysis, the DHIAP Team is developing and testing a methodology and tools for risk assessment (both expert-led and self-directed), based on the SEI's OCTAVE <sup>SM</sup> methodology. In Business Case Analysis (BCA), the DHIAP Team is analyzing operational conditions under which the Army should deploy technologies for promoting healthcare information security in its healthcare system; work includes both developing a General Methodology for conducting BCAs and applying the methodology to completing four BCA investigations. In Simulation Capability, the DHIAP Team is creating the capability to run simulations for mission survivability of defense healthcare infrastructure.				
14. SUBJECT TERMS Information Security, Information Technology, Information Assurance, RADIUS, Computer Security, Risk Analysis, Business Case Analysis, Survivable Systems, Simulation			15. NUMBER OF PAGES 288	
			16. PRICE CODE	
17. SECURITY CLASSIFICATION OF REPORT UNCLASSIFIED	18. SECURITY CLASSIFICATION OF THIS PAGE UNCLASSIFIED	19. SECURITY CLASSIFICATION OF ABSTRACT UNCLASSIFIED	20. LIMITATION OF ABSTRACT Unlimited	

NSN 7540-01-280-5500

Computer Generated

STANDARD FORM 298 (Rev 2-89)  
Prescribed by ANSI Std Z39-18  
298-102



## FOREWORD

Opinions, interpretations, conclusions, and recommendations are those of the author and are not necessarily endorsed by the U.S. Army.

( X ) Where copyrighted material is quoted, permission has been obtained to use such material.

( X ) Where material from documents designated for limited distribution is quoted, permission has been obtained to use the material.

( X ) Citations of commercial organizations and trade names in this report do not constitute an official Department of the Army endorsement or approval of the products or services of these organizations.

(X) In conducting research using animals, the investigator(s) adhered to the "Guide for the Care and Use of Laboratory Animals", prepared by the Committee on Care and Use of Laboratory Animals of the Institute of Laboratory Animal Resources , National Research Council (NIH Publication No. 86-23, Revised 1985).

(N/A) For the protection of human subjects, the investigator(s) have adhered to the policies of applicable Federal Law 32 CFR 219 and 45 CFR 46.

(N/A) In conducting research utilizing recombinant DNA technology, the investigator(s) adhered to current guidelines promulgated by the National Institute of Health.

Archie Andrews  
Principal Investigator

11/15/2000  
Date

## Table of Contents

<b>1. INTRODUCTION.....</b>	<b>5</b>
DHIAP Phase I.....	5
DHIAP Phase II .....	6
<b>2. BODY.....</b>	<b>8</b>
2.1 Technical Assessment Task.....	8
2.2 Prototype Technology Demonstration Tasks.....	9
2.3 Risk Analysis Task.....	14
2.4 Business Case Analysis Task.....	17
2.5 Simulation Capability Task.....	22
<b>3. CONCLUSIONS.....</b>	<b>25</b>
3.1 Technical Assessment Task.....	25
3.2 Prototype Technology Demonstration Tasks.....	26
3.3 Risk Analysis Task.....	27
3.4 Business Case Analysis Task.....	28
3.5 Simulation Capability Task.....	29
<b>4. APPENDICES.....</b>	<b>32</b>

## 1. Introduction

The Defense Healthcare Information Assurance Program (DHIAP) is a multi-year multi-phase program to develop and apply tools and techniques that provide an evaluation of current networks and systems and address necessary doctrine, infrastructure, training, programmatic, and technology issues to improve information assurance for Department of Defense (DoD) Medical Treatment Facilities (MTFs).

This report covers the second year of the program, from 16 October 1999 through 15 October 2000. During this period, Phase I was completed and the Phase II efforts commenced. In Phase I, the DHIAP Team evaluated vulnerabilities in current Army healthcare systems and demonstrated a prototype technology to mitigate healthcare information security threats. Phase II efforts included three areas of research: developing Risk Analysis (RA) evaluation tools and techniques and testing and refining them tools through conducting risk assessments at MTFs; developing a Business Case Analysis (BCA) methodology and testing and refining it through conducting BCAs of improved techniques for assuring protection of healthcare information in the military; and developing a simulator capability for assessing the effect of changes in security approaches of distributed systems.

### DHIAP Phase I

Phase I DHIAP activities consisted of four major tasks, each building on the results of the work that preceded it. Responsibilities of each task are summarized below.

- **Technical Assessment Task**

Using a SEI-developed methodology called "Information Security Evaluation" (ISE), the team consisting of members with experience in system security and healthcare performed on-site technical evaluations of selected military medical treatment facilities to reveal healthcare information security issues and site security exposures. Upon completion of each ISE, the team provided the site with recommendations for improving its information security posture. When other ISEs were completed, the team developed a "composite" report of findings and overall recommendations for improving protection of the military's healthcare information assets.

*Status: Initiated in DHIAP Year 1, completed in Year 2.*

- **Prototype Development Task**

Observations of the Technical Assessment task's ISEs formed the basis for the team's design, development, and testing of a technology prototype to address one or more of the identified security exposures. In addition to performing Requirements Analysis, System Selection, System Design, and Prototype Evaluation efforts, the task included an Emergent Technology Research component to develop white papers on subjects important to information protection in the military healthcare environment.

*Status: Initiated in DHIAP Year 1, completed in Year 2.*

- **Demonstration Task**

The realization of the Prototype Development task's design and testing efforts was a capability that provides demonstrable improvement to information security. The Demonstration task integrated the prototype technology into an MTF's operational

environment, applying it to a functional healthcare system. Based on results of the demonstration and lessons learned, the team produced recommendations for the policies, procedures, and methodologies required to operate the demonstration system in a safe and reliable manner.

*Status: Initiated in DHIAP Year 1, completed in Year 2.*

▪ **Technology Transition Task**

An important objective of DHIAP is to transition the technology, policies, and procedures to an operational healthcare environment. Technology Transition assures that the military MTFs involved in testing the DHIAP prototype technology received appropriate training and support for making the prototype operational and enhancing security of healthcare information at the site.

*Status: Initiated and completed in DHIAP Year 2.*

**DHIAP Phase II**

Initiation of Phase II work in April 2000 added three additional tasks to DHIAP. Responsibilities of each task are summarized below.

▪ **Risk Analysis Task**

Building on the ISE tools and techniques of the Phase I Technical Assessment task, the DHIAP Team is developing new Risk Analysis (RA) methodology and tools in association with the System Engineering Institute's Operationally Critical Threat, Asset, and Vulnerability Evaluation<sup>SM</sup> (OCTAVE<sup>SM</sup>)<sup>1</sup> project and using the new methodology to conduct a series of RAs of the healthcare information environment at selected MTFs. The first two RAs are "expert-led" by members of the DHIAP Team. Next, based on knowledge gained during those investigations, the Team adapts the methodology and tools for use in "self-directed" RAs, trains the staff at two MTFs on leading the methodology, and mentors them as they lead RAs at their own sites. The task's goal is to develop a Self-Directed Risk Assessment methodology to enable medical organizations to systematically identify assets and threats, identify and prioritize risks to those assets, and define steps for managing risk and appropriately protecting the assets.

*Status: Initiated with start of Phase II in DHIAP Year 2, active at end of Year 2*

▪ **Business Case Analysis Task**

The Business Case Analysis (BCA) task is derived from the need for the US Army to analyze the business conditions under which it should deploy technologies for promoting health information assurance in its healthcare system. In this task, the DHIAP Team first drafts a General BCA Methodology for analyzing subjects that might vary in nature (e.g., specific technologies to protect information vs. processes that use healthcare information). The team works with TATRC to select the subjects to be evaluated, and then applies the methodology to perform the BCA. As each BCA is completed, the team reevaluates the General BCA Methodology based on lessons learned and on plans for the next BCA.

*Status: Initiated with start of Phase II in DHIAP Year 2, active at end of Year 2*

<sup>1</sup> Operationally Critical Threat, Asset, and Vulnerability Evaluation and OCTAVE are service marks of Carnegie Mellon University.

- **Simulation Capability Task**

The Simulation Capability task calls for the DHIAP Team to design and demonstrate an automated simulator system that can be used to depict and analyze problems and solutions in mission survivability for Army medical systems. (A survivability simulator permits stakeholders to better understand the risks and consequences of potential threats (e.g., cyber-based attacks) to medical information systems and potentially aid in improving protection of integrity, confidentiality and availability of patient records, treatment plans, and essential medical services.) In this task, the team builds and demonstrates an alpha version of the simulator and drafts related documentation. During execution of the task, the Team obtains guidance from simulation experts enrolled as the task's Simulation Advisory Group. Dependent upon successful identification and recruitment of a DoD staff member knowledgeable in healthcare systems, the Team defines simulation scenarios relevant to Army medical infrastructure survivability. In conjunction with the Army sponsors and the Simulation Advisory group, the team prepares a plan for follow-on use of the simulator.

*Status: Initiated in DHIAP Year 1, active at end of Year 2*

## 2. Body

The work completed in the second year of the DHIAP program (16 October 1999 – 15 October 2000) includes activities that completed the DHIAP Phase I work and the efforts that initiate and accomplish initial milestones of the DHIAP Phase II work. For a summary of work performed in the first year of DHIAP, see the *DHIAP Annual Report*, 15 November 1999.

Each task outlined in the Introduction is covered individually in the material that follows. Initial comments summarizing the task's status at the start of DHIAP Year 2 are followed by a brief review of the work performed during DHIAP Year 2. An overview of the methods used in performing the Year 2 work is provided next, along with a discussion of any issues, problems, or major discoveries that arose during the time period of this Annual Report. The task review concludes with a summary of the task's results relative to goals and plans outlined in the Technical Development Plan.

### 2.1 Technical Assessment Task

As of 15 October 1999, the DHIAP Team had completed Information Security Evaluations (ISEs) at the MTFs selected to participate in the study, given each evaluated site a presentation and documentation pertaining to the evaluation's observations and recommendations, and begun analyzing the ISE data collected from the evaluated MTFs and other sources. During DHIAP's second year, the Team completed its analysis of ISE observations and recommendations, published results in the *DHIAP Phase I Composite Evaluation Report*, DHIAP Technical Report ATI IPS TR 00-02 [CER], distributed the report, and used the material as input to program and Army presentations.

#### *DHIAP Phase I Composite Evaluation Report Effort*

##### **Methods/Discussion:**

To analyze the observations and develop recommendations for the [CER], the DHIAP Team aggregated the material according to the ISE's general observation and evaluation categories and provided action recommendations to address the core security vulnerabilities. The Team noted in these recommendations that addressing the observed vulnerabilities required action at both the local level and by various higher echelon organizations.

Next, to reinforce the role of higher echelons in providing guidelines and technical support for the operational environments where the vulnerabilities exist, the Team grouped the ISE action recommendations according to certain generally recognized management responsibility areas (e.g., policy definition, procedure definition, information protection oversight, etc.). They extended the initial problem-based site recommendations to system-wide, management-focused considerations to embrace an information security culture. The initial findings presented to the MTF sites emphasized vulnerabilities that occurred at typical MTFs and required both local action and requests to outside organizations to fix. The [CER] recommendations were directed to management echelons at both headquarters and site locations and addressed how organizational structure and the overall approach to distributing and supporting medical information systems establishes an environment that can be secured at a local level or left exposed because of external dependencies.

The [CER] includes the following: overview of the processes used to select the participating military Medical Treatment Facilities (MTFs) and to conduct the information-gathering

Information Security Evaluations at those sites, an operational-oriented report of the observations and recommended actions (based on the ISE's nine technical and organizational subjects of investigation), and a management-oriented report of observations and recommended actions. Following review by the DHIAP Team and representatives of the evaluated sites, reviewer comments were incorporated into the document and it was distributed to TATRC and the sites.

**Result:** The goals of the Technical Assessment task were to develop a baseline of the current state of practice of information assurance at MTFs, develop recommendations to alleviate or eliminate identified information assurance vulnerabilities, provide immediate feedback assistance to evaluated sites, develop programmatic issues to advise and assist responsible agents within DoD, and identify organizational and technology issues for a subsequent technology demonstration effort by the DHIAP Team. The Team achieved these goals, as follows:

- Diverse organizational and technology issues were identified by the ISEs conducted at each MTF.
- Detailed feedback and recommendations were provided to each participating MTF to help them in improving the site's information protection posture.
- Analysis of information gathered during ISEs determined the types of vulnerabilities that could be common to multiple sites vs. those that were associated with situations unique to a single site.
- The vulnerability information gathered from the sites, problem-oriented recommendations, and management-focused recommendations were published in the [CER]. The report was submitted to TATRC and distributed to the evaluated MTF sites.
- Knowledge gained from completing the Technical Assessment's ISEs and developing the [CER] has been used in a variety of ways, including the following:
  - Provided input to the Requirements Analysis Effort of the Phase I Prototype Development Task (below) on information protection vulnerabilities that could be partially or completely addressed with technology solutions;
  - Provided input to DHIAP Phase II planning efforts on ways to enhance effectiveness of the Information Security Evaluation process, specifically by incorporating consideration of risks, threats, and relative value of a site's information assets; and
  - Served as the basis for presentation to the Senior Army Medical IMO September 2000 video teleconference on ISE results and systemic issues that may be encountered across many Army and DoD MTFs.

The presentation given to the Senior Army Medical IMOs is included as **Appendix A**.

## **2.2 Prototype Technology Demonstration Tasks**

The DHIAP Prototype Demonstration consisted of the Development, Demonstration, and Technology Transition efforts described below.

### **2.2.1 Prototype Development Task**

As of 15 October 1999, the DHIAP Team had completed the Requirements Analysis effort, which resulted in the decision to demonstrate a remote dial-in user authentication capability that



complied with RADIUS (RFC 2058) standards and met the Army's requirement<sup>2</sup> for authentication, control, and auditing of remote dial-in users. The System Selection effort had determined the components to be used, the System Design effort's customization and testing of prototype components was well under way, and the Prototype Evaluation effort's initial technology review by a group of system experts who were independent of the design team had been completed. In the task's Emerging Technology Research effort, the Team had performed initial research and proposed subjects for three white papers on technologies relevant to Army information protection needs. During DHIAP's second year, the Team completed the System Design and Prototype Evaluation efforts in preparation for the DHIAP Prototype Demonstration task (below) and produced and published three white papers of the Emerging Technology Research effort.

#### ***2.2.1.1 System Design Effort***

**Methods/Discussion:** The DHIAP Team completed the final, pre-rollout testing of the RADIUS-compliant prototype technology to verify remote dial-in to one site with authentication occurring at another site.

**Result:** The System Design effort produced a prototype system that was ready for operational evaluation by the DHIAP Team and staff at the participating MTFs.

#### ***2.2.1.2 Prototype Evaluation Effort***

**Methods/Discussion:** In the early months of DHIAP Year 2, the Team emulated the essential characteristics of an MTF system infrastructure in a multi-site technology laboratory environment. Installation of the prototype RADIUS-compliant system in the lab permitted aggressive testing and debugging of installation, training, and operational issues prior to the system's distribution to trial sites for installation and implementation. Since the technical infrastructure of a typical MTF consists of UNIX, NT, and VMS systems, use of the multi-site lab allowed the Team to emulate connection to and from similar type devices by configuring systems to test the existing communications protocols.

On 3 November 1999, the DHIAP Team conducted a demonstration of the prototype's capabilities for the technical staff of the MTFs that would participate in Prototype Evaluation and TATRC. The demonstration depicted connection of a dial-in user through the RADIUS-compliant prototype system to separate devices that included a UNIX system emulating a CHCS Telnet connection and a Web Server/NT Server that had been configured to represent an MTF's Exchange Web Server. Detailed instructions for installing the RADIUS-compliant technology and performing updates to the system's critical control tables were documented in the *DHIAP RADIUS Supplemental Installation and Maintenance Guide*, DHIAP Special Report IPT 00-03, April 2000 [IMG] (included as **Appendix B**). The guide would be used to train site staff for initial installation and to support the sites' ongoing maintenance of the system. The successful demonstration provided site staff with an understanding of equipment capabilities and led to their agreement to install an Initial Operating Capability (IOC) at their sites.

---

<sup>2</sup> Message, HQ Washington DC //SAIS-IAS//, subject: Network Security Improvement Program (NSIP) – Army Modem Dial-In Standards and Policy, 231300Z April 1999.



Following the demonstration of the RADIUS-compliant system, the DHIAP Team began delivering similar equipment configurations to the MTFs that would participate in the demonstration task.

**Result:** Prototype Evaluation activities produced a RADIUS-compliant prototype design that was deemed ready by the DHIAP Team and MTF technical representatives for operational testing. Installation and use of the system components in the DHIAP laboratory's distributed sites permitted the Team to iron out most installation issues prior to delivering the system to the sites. Also, as a result of performing laboratory tests, a number of potential operational issues were identified by the Team and resolved. Laboratory testing of the prototype resulted in a validation and verification of the design of the prototype's capability for identification, authentication, authorization, and accounting of remote dial-in users. The document that was drafted during Prototype Evaluation to assist sites in installing and maintaining the prototype system has since become an important resource for the sites.

At the conclusion of the Prototype Evaluation effort, the MTFs had an Initial Operational Capability which provided the sites with the opportunity to become familiar with system operation, plan for operational procedures, and plan for the migration of their user population to the new capability.

### ***2.2.1.3 Emerging Technology Research Effort***

**Methods/Discussion:** Based on knowledge of emerging information protection technology and on observations and recommendations resulting from DHIAP Information Security Evaluations at Army MTFs, the DHIAP Team proposed to TATRC in the first year of the program three areas of research pertinent to security issues of potential concern to the MTFs. The three research areas were Remote System Administration, Public Key Infrastructure, and Trust Model development. Upon receiving TATRC approval of the subjects, the Team members collaborated on research and developed the papers.

**Result:** The Emerging Technology Reports outline particular information assurance issues faced within DoD and, in particular, at Army MTFs as described below:

- *Remote System Administration: Issues and Recommendations*, ATI IPT Special Report 00-05 dated April 2000 [RSA] (included as **Appendix C-1**)

Based on a hypothesis that effective policy, operational procedures, management support, and monitoring tools would reduce the potential risks created by remote administration, this paper evaluates remote system administration in terms of DoD's actual practices, inter-agency relationships, and existing policies. It describes changes in policies and practices that would be effective for improving MTF control over external administrators, minimizing exposure of administrators' communications, and reducing exposure of other local systems if the remotely administered system should be compromised.

- *Public Key Infrastructure: Resources, Requirements, and Recommendations*, ATI IPT Special Report 00-06 dated April 2000 [PKI] (included as **Appendix C-2**)

This paper examines implications of implementing PKI in a military MTF environment. It defines the components of a typical PKI and the security requirements for those components and studies implementations in both the government and private industry current at the time of the report. In defining the general issues related to PKI deployment, it notes issues that

are peculiar to the MTF environment. Finally, the paper assesses the impact of PKI deployment and provides recommendations for initiation of a PKI pilot program in the DoD medial arena.

- *Trust Model: Defining and Applying Generic Trust Relationships in a Networked Computing Environment*, ATI IPT Special Report 00-07 dated May 2000 [TRU] (included as **Appendix C-3**)

The paper describes a trust model as a tool that helps one visualize and understand the degree of confidence that is intentionally or unintentionally granted to computer users, systems, and networks based on an understanding of associated risks that are inherent with granting this confidence. It explains how an organization gains greater awareness of threats, vulnerabilities, and the risks associated with those threats and vulnerabilities as it works to define the trust model more completely. The paper concludes that knowledge of risks allows an organization to assess each risk, determine the cost, resource availability, and/or technology for mitigating each risk, and decide whether to implement a solution to mitigate the risk or accept the risk.

### 2.2.3 Demonstration Task

In the early months of DHIAP Year 2, the Team implemented the RADIUS-compliant prototype system at MTFs, configured it to site requirements, tested it, demonstrated its functions and capabilities, prepared for transition of the prototype to the MTF sites, and obtained feedback and guidance for the DHIAP Team.

**Methods/Discussion:** The DHIAP Team began the Prototype Demonstration task by installing the RADIUS-compliant systems at the MTFs that would participate in field trials:

- The system was delivered, installed, configured, and tested at the first trial site MTF, a regional medical center, during the period of 8-10 November 1999. With the assistance of MTF IMD staff, the system was installed and initial testing was completed within the week. Site personnel who would operate and maintain the system were trained on the configuration.
- The system was delivered, installed, configured and tested at the second trial site MTF, a community hospital, in early December 1999. Installation at this site went smoothly, partly as a result of the lessons learned and experience gained from the preceding MTF installation.

The DHIAP prototype configurations installed at each of the two trial sites included full capabilities to allow thorough testing. Following system installation at the MTF trial sites, the DHIAP Team trained MTF staff on the installation, configuration, operation, and maintenance of the system. In addition, the Team reviewed the sites' policy and procedures for support of secure system operations. The MTF sites operated the RADIUS-compliant systems, configured their remote dial-in users, recommended user guidance, and monitored the remote users' activity in accessing MTF systems.

The DHIAP Team made follow-up visits to both MTF trial sites in early January to upgrade the CISCO Secure operating system to the latest version, to install the NT C2 support package, and to reinforce any training and operational issues found during the initial testing. MTF staff enhanced their operational procedures and documentation early in their work with the system and developed brief instructions for their sites' dial-in users. Based on the software upgrades and other knowledge gained during the Demonstration Task, the *DHIAP RADIUS Supplemental*

*Installation and Maintenance Guide*, Special Report IPT 00-03, April 2000 [IMG] was revised and published in final form.

**Result:** The architecture and components of the prototype developed for trials at the Army MTFs participating in the DHIAP RADIUS demonstration have proven to be suitable for use in both trial site and other military MTFs. The DHIAP prototype meets the Army's requirements for modern dial-in standards and policy in which the RADIUS standard was selected as the required method for implementing authentication, authorization and accounting.

Implementing the DHIAP technology at an MTF is a matter of installing and configuring its commercial off-the-shelf tools. The DHIAP Team and site technical staff determined that an MTF technical staff members' ability to apply prior related experience generally reduces the time and complexity of the task. By selecting components that were closely related to the technologies already installed in the trial site MTFs, the DHIAP Team was able to take advantage of the MTF Information Management staff members' existing technical expertise.

The DHIAP Team noted that the learning curve for supporting the DHIAP RADIUS-compliant system was relatively small at the regional medical center MTF where system administrators had prior experience with the Windows NT Server and Cisco IOS operating systems. In contrast, it was more time-consuming to train staff at the community hospital MTF who had no prior experience in working with routers and their software. To provide additional support where staff members did not have adequate prior experience, and where vendor documentation is inadequate, the DHIAP Team developed the [IMG]. The trial site implementations supported the DHIAP Team's expectation that a system administrator who has successfully installed one DHIAP RADIUS-compliant system would require minimal additional knowledge to install systems at additional sites (and, conversely, that an administrator with no experience in Windows NT or Cisco IOS software would not likely require outside assistance from the vendors or other experienced technicians).

#### 2.2.4 Technology Transition Task

During DHIAP's second year, the Team completed transition of the technology to the sites. The sites accepted Full Operational Capability (FOC) for the technology, trained users, and migrated users to RADIUS from prior dial-in methods. A complete report on this RADIUS architecture, including architectural alternatives to the development and the implementation of the RADIUS testbed, is contained in the *Phase I Technology Demonstration Report: Prototype for Remote Authentication Dial-In User Service (RADIUS)*, ATI IPT Technical Report 00-04 dated April 2000 [TDR].

**Methods/Discussion:** Soon after installation, the sites were able to transition the DHIAP RADIUS-compliant system from testing to full operational status. Operational use of the RADIUS-compliant technology was confirmed as the trial sites took steps to migrate the users from existing dial-in systems to RADIUS control. Upon successful test and acceptance by the site of Full Operational Capability, the DHIAP Team transferred ownership of the DHIAP RADIUS-compliant system to the control of the appropriate property book offices.

The DHIAP Team used MTF feedback on operational experience with the technology as the basis for developing recommendations for the Army's future enhancement of the DHIAP RADIUS prototype and its related policies and procedures. The design and processing of the RADIUS-compliant prototype, the Team's recommendations for future use of the technology,

and the Team's recommendations for initiating a program to implement broader use of the technology are described in [TDR]. As stated in that report, "The DHIAP prototype provides flexibility for Army facilities while supporting the Army's requirements for RADIUS-compliant systems."

**Result:** The MTF trial sites' FOC test and acceptance of the DHIAP RADIUS-compliant prototype technology occurred in February 2000. The MTF sites have briefed the facilities' Command Groups on the RADIUS installation, calling it a positive improvement in capability. Note that a Business Case Analysis evaluating usability, operational impact, cost, and level of acceptance of the RADIUS prototype is discussed in section 2.4.3 of this report.

### 2.3 Risk Analysis Task

Information assurance has become a significant concern to the medical community due to the growing amount of medical information stored in electronic formats therefore making sensitive information accessible to a larger population of networked users than ever before. Because this availability exposes the medical community to a variety of new threats that can impact on the confidentiality, integrity, and availability of the information, MTFs need a better way of understanding their information risks and creating new strategies for addressing those risks. A systematic approach to assessing information security risks and developing an appropriate protection strategy (such as the methodology developed as part of the Risk Analysis Task) can be a major component of an effective information security program. By adopting a systematic approach, an MTF can better understand its current security posture and use it to establish a benchmark for improvement.

#### 2.3.1 Design/Develop OCTAVE Tools and Methodology

**Methods/Discussion:** The Software Engineering Institute (SEI) is the principal developer of the Risk Analysis (RA) methodology. The information security risk evaluation methodology, known as OCTAVE, builds on previous work done in software engineering risk evaluation and integrating the ISE methodology used in Phase I. Based on initial prototypes of this analytical approach, the Team developed specific plans for building a methodology and tools for conducting risk evaluations in MTFs. In order to develop, test, and refine the methodology, it was decided to develop the methodology in two stages. First the prototype methodology would be used in an expert-led engagement with two MTF sites. The expertise would be provided by the SEI, reinforced as necessary by other members of the DHIAP Team. Based on lessons learned from the initial expert-led evaluations, the methodology would be refined and training developed to enable a site to perform a self-directed risk evaluation. The self-directed risk evaluation would be accomplished at two sites with the DHIAP Team that participated in the expert-led risk evaluations training, observing, mentoring, and reinforcing as requested. Key to successful development and refinement of the methodology was recruitment of the four MTF sites for participation in the DHIAP Phase II Risk Analysis activities. The Team worked closely with TATRC to recruit MTFs from each of the military branches to gather experience and help ensure development of a robust methodology.

The DHIAP Team began their development of the RA methodology by outlining the major steps of the assessment process, from initial interviews for gathering data about information assets and potential threats, through operational and technical analysis of that data, to prioritizing information assets and determining appropriate actions for protecting the assets. The Team then

added structure by defining the necessary elements of each step of the process, identifying the type of site staff who should participate, and estimating the amount of time needed to perform each of the steps. Going into further detail, the Team drafted materials to be used in each step of the process—both presentations describing the work to be done and worksheets for the Team and site participants to use in performing that work. In doing all of this work, the Team conducted numerous design meetings and internal reviews of the materials they were creating and updated the RA methodology materials/worksheets/templates as appropriate. A technical assessment of the draft RA methodology was conducted on 9 August 2000 to gather additional ideas and input from DHIAP Team members, and a formal review was held on 22-23 August 2000 to assure the methodology's readiness for use in the field.

**Result:** The DHIAP Team developed methodology and tools for identifying and managing information security risks using a series of expert-led workshops. The methodology supports identification of information assets important to the mission of the organization, threats to those assets, and vulnerabilities that may expose the assets to threats. Additionally, it organizes the processes of prioritizing the information assets and developing strategies for protecting them. For each of the planned workshops, the DHIAP Team created presentation materials that establishes the purpose of the workshop and focuses the workshop's information-gathering and decision-making activities. Templates and worksheets to be used by the session leader and participants during the workshop, and notebook pages describing how the session leader should facilitate the workshop, were also developed.

Experience gained from working with these materials in the initial expert-led risk evaluation allowed the DHIAP Team to enhance and revise them. In addition, observations were captured to support the Team's upcoming task to adapt the materials and facilitation instructions for use in a site-led (self-directed) Risk Analysis effort.

### ***2.3.2 Recruit/Select Sites for OCTAVE-based Risk Assessment***

Concurrent with methodology development, TATRC and DHIAP Team members worked to identify and recruit sites to participate in the four risk evaluations to be conducted during DHIAP Phase II. It was the group's preference that facilities of differing sizes and from each branch of the military service be included among the participating MTF sites. DHIAP plans called for the first two risk evaluations to be expert-led, meaning that the DHIAP Team would guide the RA process with the MTF staff. Following delivery of the first two expert-led risk evaluations, the OCTAVE methodology would be refined and OCTAVE training developed, and the second set of MTF sites trained to execute the OCTAVE methodology in a self-directed mode (i.e., the MTF staff executing the process, obtaining outside assistance as necessary to provide expertise the sites lack).

The Risk Analysis Team worked closely with TATRC to recruit MTF sites for participation in the Risk Analysis efforts. The process of recruiting sites from each of the service branches required significant coordination and assistance from the services' medical Chief Information Officers to identify likely sites to participate in the expert-led risk evaluations. To date, an Air Force site has been engaged for the first expert-led risk evaluation and a Navy site has been identified for the second. Because of a strong continuing relationship with the DHIAP Team, the two MTF sites that participated in DHIAP Phase I readily agreed to participate in Phase II's self-directed risk evaluations, providing the opportunity to compare and contrast the execution of an outsider-led evaluation (the ISE process) with a self-directed risk evaluation executed



predominantly by internal resources. An additional Army site identified and contacted as a potential site for either the expert-led risk evaluation or for a self-directed risk evaluation has expressed interest in participating, either in the near term or as an early participant in future deployment of the risk evaluation methodology.

**Result:** Four sites have been recruited to participate in the DHIAP Risk Analyses. The expert-led risk evaluation at the Air Force site is in progress at the time of this report. A second expert-led risk evaluation at the Navy site is scheduled to begin following completion of the first evaluation. The two Army sites that had participated in DHIAP Phase I agreed to participate in DHIAP Risk Analysis training and conduct self-directed risk evaluations. In addition, a fifth MTF site (Army) expressed interest in performing a Risk Analysis, either after the conclusion of the currently scheduled evaluations or in lieu of one of the committed MTFs.

### ***2.3.3 Conduct DHIAP-Led Risk Assessments at Two MTFs***

In preparation for using the expert-led risk evaluation methodology, the DHIAP Team met on 22-23 August 2000 to conduct a final Team review and walk-through and to prepare all participants for the upcoming event. The first evaluation activity, the Senior Management Workshop, was conducted at the first site on 6 September 2000. This workshop was held in two parts. Part one, attended by the site's senior staff, explained the goals of the risk evaluation process and the role of the site's staff in each step of the process; in addition, it provided an overview of the schedule and resource requirements for conducting the risk evaluation at the site. (The presentation used in this session is included as **Appendix G**.) Part two of the 6 September meeting was a discussion of the operational areas that the site's senior staff wanted to investigate in the risk evaluation and their reasons for naming these areas. The meeting concluded with the MTF staff selecting a site coordinator and specifying operational management personnel who, based on their knowledge of the areas to be investigated, were designated to participate in the evaluation.

The DHIAP RA Team worked with the site coordinator to schedule the next group of workshops at the site. Based on MTF staff availability, 26-28 September 2000 was selected as the dates for the next workshops. At that time, the DHIAP Team conducted an Operational Area Management Workshop and a Staff Workshop with hospital clinical and administrative personnel to identify key assets and their values, threats to those assets, indicators of risk, and the type of protection strategy currently employed by the site. Next, applying an approach that was similar, but more technically focused, the Team conducted a Staff Workshop with the medical center's information management staff to obtain input on the site's key technical assets and their values, their threats, indicators of risk, and current protection strategies. At the conclusion of the visit, the Team leader worked with the site coordinator to identify individuals from the preceding workshops who would participate throughout the rest of the RA process as part of the risk evaluation's Analysis Team and to schedule the next visit.

On 10 October 2000, the DHIAP Team returned to the MTF to meet with the Analysis Team and conduct the Security Requirements Workshop. In that workshop the DHIAP Team led the site's Analysis Team through a process of selecting the site's most critical information assets and the security requirements for each and then defining threats by developing threat profiles. This meeting concluded the process of documenting the RA methodology's Organizational View of information assets and threats at the facility. On 11 October 2000 the DHIAP Team, meeting with the site's technical staff, conducted the IT Protection Strategy Workshop to map high

priority information assets to the site's information infrastructure. This effort entailed mapping the critical assets to classes of systems and identifying the access/data paths; it culminated with the technical identification of the site's key assets. Vulnerability scans, a planned event of the technical analysis, were not performed due to local restrictions on access to the site's networks, workstations, and servers. In lieu of an extensive network scan and observation of workstation and server settings, the Team was provided with an analysis report compiled by the communications agency supporting the MTF based on a sampling of the MTF systems. Although the restrictions prevented a fuller vulnerability evaluation, the DHIAP Team's ability to assist the site in developing strategies for security risk management and asset protection will not be noticeably impacted. In concluding this visit, the DHIAP Team scheduled the remaining workshops of the site's expert-led risk evaluation, with the work scheduled for completion in mid-November 2000.<sup>3</sup>

**Result:** The first of the two expert-led risk evaluations has proceeded smoothly. Senior and operational staff have expressed great interest in the process (for the present and for ongoing use) and eagerness to see and work with the results. Each workshop process has been conducted with appropriate site representation, executed smoothly, and elicited valuable information regarding the MTF site for the succeeding workshop's process. In addition, experience gained from conducting and participating in the workshops has been captured in a useful format for use in improving the OCTAVE methodology and its tools and in migrating the process from its current expert-led orientation to the self-directed approach required later in this DHIAP effort.

## **2.4 Business Case Analysis Task**

The goals of the Business Case Analysis task are to analyze the business conditions affecting the deployment of technologies supporting information assurance in the military healthcare systems and to develop a business case methodology to support future analysis of potential information assurance technologies. As appropriate to the purpose of the study, the studies and the methodology include an assessment of feasibility, cost, benefit, and availability of information assurance technologies in access control, authentication, file integrity, and/or data transfer.

### **2.4.1 Develop BCA Methodology/Metrics**

**Methods/Discussion:** The DHIAP Team met on 19-20 April 2000 to begin a process of defining the overall methodology and metrics for performing a Business Case Analysis (BCA) of information protection techniques. The initial draft of the BCA Methodology was based on the professional experience of the Team members and a variety of documented business analysis techniques compiled in advance of the meeting. The initial approach included the following steps: planning; gathering background information; refining the project's scope, plan, and schedule; determining how to best collect the data (sources, forms, interviews, etc.) and organize what was collected; collecting, compiling, and analyzing the data; developing conclusions; and producing the report. The Team recognized that, in order to make the BCA Methodology useful and unique, they should define it in such a way that it would be applicable across a broad range of topics, from evaluation of a technology to investigation of an organizational practice or process.

---

<sup>3</sup> The last workshops of the series were completed after the closing date of this report and therefore will not be included in this report.

The development of the BCA Methodology was planned to be an iterative refinement effort (i.e., develop criteria, test the generality and appropriateness of the criteria by application to a BCA, and then refine the methodology based on lessons learned during each BCA). The Team identified for the initial draft of the methodology the four evaluation categories that might be appropriate to BCAs on diverse subjects: cost factors/resource requirements, non-cost/functional/operational factors, risk factors, and economic analysis factors. They also developed a number of specific evaluation criteria for each category, along with a brief definition for each one.

During another segment of the meeting that focused on developing candidates for investigation in upcoming BCAs (see "Select BCA Subjects and Sites" below), the Team refined the draft methodology each time the topics under consideration suggested additional process steps, evaluation criteria, etc.

**Result:** The DHIAP Team presented the initial version of the BCA Methodology to TATRC on 15 May 2000 (the presentation is included as **Appendix D**), and refined the methodology based on guidance received at the meeting. The initial version of the draft BCA Methodology was used to define the plan and manage the execution of the first DHIAP BCA on DHIAP's Phase I RADIUS Implementation. Then, based on lessons learned from the RADIUS BCA and new information gathered while planning for the second DHIAP BCA, the methodology was further refined. The methodology will undergo continuous refinement based on lessons learned in the conduct of the BCAs.

#### **2.4.2 Select BCA Subjects and Sites**

**Methods/Discussion:** During the DHIAP Phase II Kickoff Meeting held 16-18 February 2000, TATRC and the DHIAP Team agreed that the first BCA topic to be examined using the DHIAP developed BCA Methodology would be the RADIUS-compliant systems deployed at two military MTFs during DHIAP Phase I. It was also agreed that the topics of BCAs #2, 3, and 4 would be selected by TATRC following a process where the DHIAP Team identified candidate topics and received guidance on the relative importance of each as a BCA subject.

In its 19-20 April 2000 meeting, the Team developed a process for defining and selecting subjects most appropriate for study by a business case analysis. Major steps of the approach are the following:

- **Derive Candidate Topics:** Based on the expressed needs of the intended audience for the report, identify potential topics of investigation. Depending on the subject, it may also be important to clarify the scope of the topics.
- **Determine Candidate Screening Criteria:** Determine criteria to be used for screening the candidate topics based on the customer's major drivers. Screening criteria might include such subjects as customer priorities, cost/resources/timeframe/expertise needed to examine the subject, potential benefit (e.g., operational improvement, regulatory compliance, risk reduction) resulting from performing the study, breadth of applicability of the study's results, and existence of an advocate/champion/sponsor for the effort.
- **Evaluate the Candidates:** Build a matrix similar to the one included as **Figure 1**, listing Candidate Screening Criteria in the first column and creating empty columns for each Candidate BCA Topic. Consider the importance of each Candidate Screening Criteria element to selection of the topic (e.g., Do particular customer priorities reign as the most



important consideration in selecting the topic?), and assign a "Weight" to each criteria element. Next, focusing on one candidate BCA Topic at a time, define a "Raw Score" to indicate how well the Topic satisfies the Criteria element. Complete the matrix by computing each BCA topic's "Weighted Total" (i.e., by Topic, multiply each criteria element's "Weight" by "Raw Score" to produce its "Weighted Scores," then compute the sum of all "Weighted Scores").

- **Select BCA Topic(s):** The "Weighted Total" of each candidate BCA topic provides an indicator of its importance relative to the other topics. It is not unusual to be surprised by

some results when evaluating the topics' weighted total scores. In studying how certain unexpected results were produced, the evaluators may realize that they have over- or under-rated the importance (weight) of certain candidate screening criteria. Or, they may realize that they should alter the raw scores assigned to certain topics/criteria. These are valid reactions, ones that serve to improve the quality of assessments made through the table. The process of assigning weights and scores forces evaluators to stand back, take a different and less biased look at both the criteria and their applicability to the topics; it may even lead to adding/changing/deleting certain topics. While not conclusive, the approach of developing weighted total scores provides a less biased way of evaluating the topics and is a form of sanity check on the process of selecting one topic over others.

The DHIAP Team developed a list of potential topics for BCAs and then, using the process outlined above to refine and prioritize candidate topics for BCAs #2 – 4, recommended priority topics to TATRC. Based on guidance received from TATRC, the DHIAP Team presented the selection process (see above) and the weighted scores produced for the priority topics.

**Result:** The candidate BCA topics presented by the DHIAP Team at the 1 June 2000 DHIAP Interim Progress Review (the presentation is included as **Appendix E**) were: User Friendly Authentication, Role-Based Access Control, and Network Auditability. These proposed BCA topics were approved by TATRC as subjects for BCAs # 2-4. The methodology used to select these topics from a much longer list of candidates is outlined above.

#### 2.4.3 Conduct BCA #1: DHIAP Phase I RADIUS Implementation at Military MTFs

**Methods/Discussion:** The purpose of BCA #1, DHIAP RADIUS Implementation, was to provide an analysis of the operational factors related to installing a RADIUS-compliant capability for securing remote dial-in access to information systems at military MTF sites. In their meeting on 19-20 April 2000, the DHIAP Team used the draft BCA Methodology to develop the plan for conducting the RADIUS BCA. The Team enhanced and revised portions of the draft methodology as their RADIUS BCA planning provided new perspectives and posed

Candidate Screening Criteria	Weight (0-10)	Candidate BCAs					
		Topic 1		Topic 2		Topic 3	
		Raw Score	Weighted Score	Raw Score	Weighted Score	Raw Score	Weighted Score
Customer Priorities	10	10	100	6	60	4	40
Team Expertise	8	8	64	8	64	9	72
Potential Benefit	7	8	56	9	63	6	42
Compliance	6	9	54	9	54	8	48
Risks	5	7	35	5	25	7	35
Breadth of Applicability	4	9	36	8	32	9	36
Champion	3	8	24	9	27	7	21
<b>WEIGHTED TOTAL:</b>		<b>369</b>		<b>325</b>		<b>294</b>	

Figure 1 – "Weighted Score" Approach to Evaluation of Candidate BCAs

situations and opportunities not yet addressed by the initial draft of the methodology. The draft methodology presented to TATRC on 15 May 2000 included results of the RADIUS planning updates. BCA #1 planning began with establishing the scope of the investigation which included an analysis of the operational factors, costs, risks, and benefits related to installing a RADIUS-compliant capability for securing remote dial-in access to MTF information systems.

Cooperating with TATRC on site recruitment visits and conference calls, the Team obtained commitment to participate from the two military MTFs who had participated in the DHIAP Phase I implementation and demonstration of a RADIUS-compliant prototype system. Major tasks of the DHIAP Team's RADIUS BCA effort included the following:

**Table 1 – Major Tasks of the RADIUS BCA Effort**

1. Establish BCA Scope	6. Identify Data Collection Tools
2. Collect Background Information	7. Organize the Data Collection Process
3. Define BCA Plan and Schedule	8. Collect the Data
4. Develop and Refine Evaluation Criteria	9. Analyze the Data
5. Identify Data Sources	10. Develop Conclusions and Report

The bullets that follow summarize how the DHIAP Team executed the key process steps of the RADIUS BCA.

▪ **Establish BCA Scope, Collect Background Information, Define Plan and Schedule**

The Team established a three-part scope for the RADIUS BCA as follows:

1. Examine the vulnerabilities and requirements that led to the decision to implement a RADIUS-compliant capability at two Army MTFs in DHIAP Phase I in December 1999;
2. Consider the costs and risks of these implementations; and
3. Compare cost and non-cost factors at each MTF before, during, and after the RADIUS implementation.

The Team gathered background material on RADIUS-related technologies and their use from the Internet and other commercial sources and on its implementation at the demonstration sites from existing DHIAP materials. Next, armed with relevant information, they planned their approach to performing research and reporting the results.

▪ **Develop Evaluation Criteria, Identify Data Sources and Data Collection Tools, and Organize the Data Collection Process**

The Team determined the most appropriate data sources and collection tools for obtaining the methodology's cost factor and non-cost factor data within each evaluation category. It is worth noting that one of the unique qualities of this BCA is that the analysis could be accomplished with the benefit of hindsight: Because the RADIUS-compliant demonstration systems were already installed and operational at the two MTFs, data could be obtained with a minimum of speculation for the pre-, during, and post-implementation timeframes. For RADIUS conversion costs, the primary source of data was the accounting records of the DHIAP Phase I participants; for pre- and post-RADIUS sustainment costs, the Team relied heavily on the knowledge of the Information Management Office (IMO) at each MTF.

To organize the data collection process, the DHIAP Team developed several types of survey forms, as follows:

- Cost collection sheet for use in the interview session with the IMO;
- Questionnaires to gather non-cost information from the IMO and the Information Management Division (IMD) staff; and
- Survey forms that could be answered online via email for users of the RADIUS remote dial-in support.

▪ **Collect Data**

The Team visited the demonstration sites on 11-12 July 2000, spending about a half day at each one. They first interviewed the site's IMO, using the IMO Interview Guide, then the IMD staff participated in an interview that was based on the IMD Technical Interview Guide. In the IMO interview, the Team made final plans for how each site would deliver the User Questionnaire to its users and obtain their responses. Since both sites had email groups established for their dial-in users, both sites determined that the site IMD staff would distribute the User Questionnaires, perform follow-up to obtain responses, and forward completed questionnaires to the DHIAP Team. The Team concluded the BCA data collection activity by merging all interview notes and the results of User Questionnaires into data repositories designed to support analysis. Note that the interview guides, and the questionnaires are included in the *DHIAP RADIUS Implementation Business Case Analysis*, Technical Report ATI IPT TR 00-08 dated October 2000 [RAD].

▪ **Analyze Data, Develop Conclusions, and Develop Report**

The Team distributed responsibility for developing the initial data analyses based on the individual Team members' areas of expertise, and then worked jointly on drafts of the report to produce the BCA report document.

Prior to publication of the [RAD] report, the report was provided to the sites and TATRC for review and comment. The Team addressed questions resulting from the review and incorporated inputs to produce the final report.

**Result:** The [RAD] reports that the key benefits derived from installing RADIUS included information protection, control over dial-in user access, and regulatory compliance. Based on the analysis, the DHIAP Team concluded, "a properly managed RADIUS implementation is an effective and desirable approach to controlling remote user dial-in access."

#### **2.4.4 Conduct BCA #2: Effective Authentication in a Medical Environment**

**Methods/Discussion:** The general subject of "User Friendly Authentication" was agreed for BCA #2 at the DHIAP IPR on 1 June 2000, along with guidance from TATRC that a more refined definition of the subject should be submitted for TATRC approval before BCA #2 work was begun. As the DHIAP Team began planning for BCA #2 in a meeting on 7-8 September 2000, they evaluated a number of potential subject refinements and developed the following scope: "In the context of MTFs, analyze capabilities and limitations of various authentication technologies in light of MTF situational dependencies, and present results and indicative costs for acquisition decision makers." The DHIAP Team concluded that the technologies for user friendly authentication (including SmartCards and a wide variety of biometric devices) are

rapidly evolving, changing in both capability and price. Since many organizations are evaluating these types of technology, a DHIAP Team investigation of the actual tools would be a duplicative effort providing information that would quickly be out of date.

Following discussion with TATRC, it was agreed that it would be more useful for the Army and DoD if the DHIAP BCA Team study the unique circumstances at an MTF that could make the rollout of authentication technologies difficult or even impossible in that setting. Accordingly, the "User Friendly Authentication" title of the BCA was changed to "Effective Authentication in a Medical Environment."

As the Team developed the detailed plans for conducting the BCA, they used lessons learned from conducting BCA #1 to improve the quality of their process and revise the BCA Methodology accordingly. The BCA Team then drafted the scope, evaluation criteria, and report outline for BCA #2 (included as **Appendix H**).

Since the September meeting, the Team has completed their research assignments and developed components of the draft report. The Team's research has included gathering data on MTF staff members' access to and use of computer systems from other DHIAP sources, including: the Phase I ISEs at two MTF sites, Phase I's implementation of the RADIUS-compliant prototype at the same sites, and the Phase II Risk Analysis being performed concurrent with this BCA.

**Result:** A meeting to review the draft report for BCA #2 and complete its development was scheduled for 18-20 October 2000; schedules call for the report to be distributed to TATRC for review and comment in November 2000.

#### **2.4.5 Conduct BCAs #3 (Role-Based Access Control) and #4 (Network Auditability)**

**Methods/Discussion:** Preliminary descriptions of the scope for BCA #3, Role-Based Access Control, and BCA #4, Network Auditability were developed as a result of discussions during the 7-8 September Team meeting (included as **Appendices I and J**, respectively). Preliminary schedules call for BCA's #3 - 4 to be conducted in parallel, and to be completed in March 2000.

**Result:** Preliminary statements of scope were submitted to TATRC for review and comment.

### **2.5 Simulation Capability Task**

The survivability simulation system is part of an existing long-term research and development effort at the Software Engineering Institute to better understand and develop more effective methods for addressing survivability issues in networked systems, with particular emphasis on critical national infrastructures including military healthcare infrastructure. Survivability research seeks new techniques to analyze essential requirements for and potential impacts on mission fulfillment in unbounded systems. Easel is an automated simulation tool for research on survivability, infrastructure assurance, and other applications that must content with incomplete and imprecise information. A background whitepaper describing survivability and the Easel tool, *Survivability and Simulation* by David Fisher, is attached as **Appendix L**. The technical approach to the simulation development includes:

- Design of a discrete event simulation programming language;
- Implementation of a translator, a run-time system, and a visualization system for that language; and
- Development of documentation for simulation authors and users.

The preliminary functional version (i.e., alpha release) of the survivability simulation system will be demonstrable during the DHIAP Phase II period of performance; it will enable test use and demonstration capability of the survivability simulation system within the SEI facility.

As of 15 October 1999, the DHIAP Team had completed significant work on the theory and formal characterization of survivability, unbounded systems, emergent algorithms, and survivability architectures. Design of the Easel simulation language and run-time environment was an active task, and several applications were under development or being studied for use of the simulator.

### ***2.5.1 Develop/Demonstrate Survivability Simulator***

**Methods/Discussion:** Development work on the simulator began with the definition of revisions to design work done under other funding. The revisions were then implemented in the lexical analyzer and parser. Other early work on the simulator included developing the first versions of the semantic analyzer and code generator, developing an interpreter that uses a structured representation of programs, and testing the interpreter on simple cases. During June - August 2000 when nine full-time students were available to work on the simulator, development activities focused on the translator. By the end of August 2000, the software testing had proven that Easel programs could be translated and executed.

**Result:** The alpha version of the simulator is currently on schedule for initial testing in March 2000 and a demonstration for TATRC at the SEI facilities by the end of March 2000.

### ***2.5.2 Provide Preliminary Manual and Guide***

**Methods/Discussion:** The Team began reviewing and developing the Easel Language Reference Manual (ELRM) and the Easel Author Guide (EAG). Easel is the high-level programming language that will be used to develop the automated simulation system.

**ELRM:** Much of the detailed information for the ELRM was written during the period, and the preliminary version of the ELRM was published in July 2000, with further revision as version 1.0.3.3 [ELR], attached as **Appendix L**. Further refinements of the ELRM will be necessary as the Easel Language itself matures.

**EAG:** Experimentation has shown the Team that the EAG should be built around examples. Most of the EAG development work has focused on developing and refining these examples.

**Result:** Development of the ELRM will continue, and a preliminary version of the EAG is on schedule for delivery in January 2001.

### ***2.5.3 Coordinate with Advisory Groups and Conduct Technical Meetings***

**Methods/Discussion:** A Simulation Advisory Group (SAG) comprised of DHIAP Team members, TATRC, Army simulation advisors, and subject matter experts (SMEs) is to be established to develop a plan for use of the simulation system in Army medical systems. The Team leader made some valuable contacts while attending the Defense Modeling and Simulation Office (DMSO)-sponsored conference in late May 2000. Planning for the SAG with TATRC and potential participants from the Simulation Training and Instrumentation Command (STRICOM) and the Army Research Laboratory is ongoing.

A draft charter for the Simulation Advisory Group that outlines the group's purpose and desired member characteristics was developed. The DHIAP Team members reviewed the document and provided comments, and conducted a conference call to identify potential SAG members and plan for recruitment.

SEI and TATRC have agreed that having an Army person working at SEI on the simulation effort would be advantageous for the project. SEI developed an outline of the experience and educational requirements that would be desirable in such an individual, and provided it to TATRC; however, locating such an individual who meets the desired qualifications and is available has been difficult and has produced no tangible result to date.

**Result:** Efforts to recruit SAG members have not produced results as quickly as planned, so the SAG Kickoff Meeting has been deferred, tentatively scheduled for January 2001.



### 3. Conclusions

The DHIAP activities performed in Year 2 have resulted in a number of tangible improvements to the security of military healthcare information.

#### 3.1 Technical Assessment Task

- The *DHIAP Phase I Composite Evaluation Report*, ATI IPS 00-02, February 2000 [CER] of the aggregate findings of Phase 1 Information Security Evaluations provides useful information on at least two levels:
  - Specific actions to address reported vulnerabilities; and
  - Programmatic recommendations for higher echelons to consider for implementation in order to preclude or reduce risks to information confidentiality and integrity across the DoD's MTFs.
- The vulnerabilities identified in the [CER] identified organization and technology issues that should be considered as potential vulnerabilities by other MTFs and other healthcare organizations throughout DoD.
- When results of the Technical Assessments were briefed to Senior Army Medical IMOs, they expressed strong interest in the DHIAP Team's depiction of systemic issues in protecting healthcare information in the military. As a result of the brief, they requested that they be briefed on DHIAP activities and interim results on a quarterly cycle.

The [CER] and the site evaluations on which it is based were designed to identify areas of exposure for healthcare information at an MTF and define strategies and approaches to eliminating or reducing the exposure. While each ISE focused on an MTF site and the policies, procedures, and practices at that site, the process of analyzing why certain exposures exist and developing recommendations for reducing those exposures indicated clearly that responsibility for defining and implementing solutions includes organizations beyond the MTF—in the Army's and DoD's organizations responsible for policy and system selection and support.

The ISEs identified a wide range of vulnerabilities that could impact information assurance at the MTF; they also provided specific recommendations to address each issue. The participating MTFs took action to address many reported issues, where the activities required to resolve the vulnerability were within the responsibility, authority, and capability of the MTF. However, the site leaders' and site staff members' reactions to exit briefs at the end of the ISEs highlighted the MTF's dependency on outside organizations in order to implement many types of changes that could improve local security. For example, a number of technical vulnerabilities identified in the MTF's application systems had to be reported to those centralized support organizations for corrective action. Further, the site was not necessarily notified of a fix being implemented; staff would have to re-test the system in order to learn whether the fix had been applied. For these vulnerabilities, the actions required to accomplish change are beyond the authority of the individual MTF and require decisions and actions by higher echelons.

The [CER] points out that a missing element in the military healthcare environment is a comprehensive security strategy for health information assurance that addresses each operating

level within the military medical domain, from the MTF to the MEDCOM. It recommends that such a strategy be built on:

- Clear Security Vision: Recognition of the risks that organizations composing the enterprise can tolerate;
- Commitment: Senior management buy-in and resources to execute the strategy;
- Training: Implementing security as part of the normal operations at all levels; and
- Accountability: Clear delineation of who carries responsibility for doing what and a mechanism to measure progress.

It is important to note that the importance of having a meaningful security strategy increases as emerging regulatory and legislative guidance place increasing emphasis on information security and privacy. In addition, pending regulatory guidance driven by the HIPAA legislation will cause increased attention to information assurance on the part of accrediting bodies such as JCAHO.<sup>4</sup> Implementing the recommendations included in the [CER] will establish a foundation for future efforts to comply with the forthcoming laws and regulations. The report's management recommendations (summarized under headings of information protection oversight, policy, technology standards, procedures, organizational responsibility/authority for security, and technology) outline the type of support and guidance needed from the echelons above the individual MTFs.

### 3.2 Prototype Technology Demonstration Tasks

- MTF implementation of the RADIUS-compliant technology was the basis for improving the MTFs' policy and operational procedures relative to remote user dial-in access. Changes implemented by one site or the other included:
  - Defined the access permissions of MTF dial-in users more restrictively than in the past, in some cases precluding their dial-in attempts from gaining any level of access at all;
  - Prevented dial-in users from accessing the Internet by way of MTF resources;
  - Searched for and removed accessibility from stand-alone modems that existed on some MTF desktop computers;
  - Use the RADIUS prototype's audit logs to monitor dial-in access activity;
  - Rapidly established system access during emergency (e.g., loss of Army communication lines to remote clinic); and
  - Rapidly established system access for a new user (e.g., a new mobile MRI van).
- The Emerging Technology Research papers on *Remote System Administration*, *Public Key Infrastructure*, and *Trust Model* provide assessments of technical/operational subjects of potential importance to MTFs and DoD healthcare.
- The Demonstration of the RADIUS-compliant technology established validity of the DHIAP equipment configuration and operational procedures for providing the DoD-mandated

---

<sup>4</sup> JCAHO is the Joint Commission on Accreditation of Healthcare Organizations



compliance with the RADIUS standard; the configurations minimized impact on the MTFs by using hardware and software compatible with already installed systems.

- The installation and operation guide (*DHIAP RADIUS Supplemental Installation and Maintenance Guide*, ATI Special Report IPS 00-03) developed by the DHIAP Team has provided the trial site MTFs with first-line support for maintaining their DHIAP RADIUS-compliant systems.
- The after-action report/lessons learned document (*DHIAP Phase I Technology Demonstration Report*, ATI IPT TR 00-04) provides decision makers a detailed explanation of RADIUS implementation alternatives for evaluation as a possible standard for providing improved dial-in user support and control at the MTF and the various regions.
- The DHIAP Team has continued to provide assistance to staff of the Phase I trial site MTFs as they request help with operational issues arising from operation of their RADIUS-compliant system configurations.
- The multi-site lab environment implemented by the DHIAP Team, which allowed the team to rapidly test the system, work through bugs in equipment and documentation, and emulate many device types for successful prototype testing and evaluation prior to demonstration, remains in place for ongoing support of the installed MTFs—testing compatibility of new software releases, emulating and determining fixes for operational problems reported by the installed MTFs, etc.

The demonstration of DHIAP's prototype RADIUS-compliant system showed: the ease of implementing it in the Army's existing regional network environment; its ability to work within the regions' and sites' Windows NT-based technical environment; and, most importantly, its effectiveness in providing RADIUS compliance for remote dial-in users of military healthcare systems. Its overall usefulness and fit with the existing environment are indicated by the demonstration sites' continued use of the prototype to control access by remote dial-in users even following completion of the DHIAP trials.

Experience gained by the DHIAP Team in developing and implementing the RADIUS-compliant system within an Army region at large and small MTFs gave them relevant background for providing high-level recommendations for the program management and programmatics of a broader implementation of the DHIAP RADIUS-compliant prototype across Army MTF sites.

### 3.3 Risk Analysis Task

- The DHIAP Team is applying well-developed risk identification and assessment techniques with information security evaluation techniques and practices to create the information security risk evaluation methodology/toolset known as OCTAVE.
- Sites with widely varying characteristics have been recruited for the testing and enhancement of the DHIAP Risk Analysis process, as follows:
  - Expert-Led Risk Evaluations
    - An Air Force facility that is a small MTF
    - A Navy facility that is a very large and complicated MTF

- Self-Directed Risk Evaluations

- An Army facility that is a medium-size community hospital MTF
- An Army facility that is a large regional medical center MTF

Commencement of site Risk Analysis efforts had been somewhat delayed due to the extended time required to recruit and schedule a first site to participate in the effort. However, because the DHIAP Team was able to use this time to complete, test, and enhance the Risk Analysis methodology, there was an unanticipated benefit of conducting an initial expert-led risk evaluation that was smooth, efficient, and relatively unobtrusive to the site.

It is interesting to note that, while there is great diversity in the size and mission of the institutions, the Risk Evaluation at the very small Air Force MTF is producing observations about many of the same information vulnerabilities and policy/procedure shortcomings that were noted in the Phase I ISEs at the Army's community hospital and regional medical center. In addition, there are other observations that seem, preliminarily, to be attributable to a difference in the technology support approach taken by the two service branches. These types of findings are establishing a rich base of information for an end-of-Phase II composite analysis of RA results.

### **3.4 Business Case Analysis Task**

#### **3.4.1 *Develop BCA Methodology/Metrics***

- The draft BCA Methodology for assessing the effectiveness of technology/processes to promote health information assurance is "work in progress." The initial version and subsequent refinements have been "tested" by guiding the efforts of two Business Case Analyses. The methodology is proving to be relatively easy to use.
- Its initial draft, which was particularly appropriate for analyzing the case of technology implementation because it took a before-during-after view, produced the RADIUS Implementation BCA. Lessons learned from conducting the first BCA led the DHIAP Team to alter the list of activities involved in conducting the analysis. Looking back, the Team determined that the initial draft had prescribed the right series of activities. However, some that were subtasks should have been defined as stand-alone activities, and some stand-alone activities should have been grouped together under the heading of a single major activity.
- The topic of the second BCA, a study of environment/processes and the technologies that would best fit them, drove the team to make certain enhancements to evaluation criteria. A number of criteria were added, particularly to the "non-cost factors" list. Also, instances of minor confusion in the meaning of certain evaluation criteria were cleared up. Interestingly, the activities of conducting the BCA remained fundamentally the same.

#### **3.4.2 *Select BCA Subjects and Sites***

The subject of the first BCA, DHIAP RADIUS Implementation, was agreed upon at the Phase II Kickoff Meeting in February 2000, and all agreed that the sites that had participated in the Phase I RADIUS demonstration would be the data sources for the BCA. Since then, the DHIAP Team has worked closely with TATRC to define subjects of BCAs #2-4. The subjects agreed upon (User-Friendly Authentication, Role-Based Access Control, and Network Auditability) are expected to exercise the methodology development activity to a great extent while also providing

useful information to the Army and DoD about appropriateness of specific information protection approaches and issues.

### **3.4.3 Conduct BCA #1: DHIAP Phase I RADIUS Implementation at Military MTFs**

- The first BCA, [RAD], assessed the installation of a RADIUS-compliant system as an effective and desirable approach for information protection, dial-in user control, and regulatory compliance. Key benefits derived from installing RADIUS included:
  - Information Protection. Protection of MTF information is enhanced.
    - RADIUS permits the MTF Information Management Division (IMD) staff to enforce policies regarding dial-in user authentication and authorization.
    - RADIUS improves dial-in user authentication and authorization, therefore regulating potential access to MTF systems.
  - Dial-In User Control. MTF IMD staff can now explicitly control dial-in user access to MTF systems and resources.
    - Control over dial-in access to MTF information resources resides with the MTF, whose IMD staff is appropriately equipped to know who should be allowed access and to what resources.
    - RADIUS restricts user access to only those system resources specifically permitted, while being relatively transparent to operational users and having no real impact on availability, reliability, and communications speed.
  - Regulatory Compliance. By introducing an information protection strategy based on good information security practice, the MTFs attained compliance with Army DISC4 guidance and improved their readiness to comply with emerging HIPAA, JCAHO, and NCQA<sup>5</sup> requirements.

### **3.4.4 Conduct BCAs #2-4**

- The second BCA, initially called User-Friendly Authentication, and subsequently changed to Effective Authentication in a Medical Environment, is currently in progress. Observations made and information gathered by the DHIAP Team when conducting Phase I ISEs, the Phase I RADIUS development and implementation effort, and the first Phase II Risk Evaluation have provided useful background material for this BCA.

## **3.5 Simulation Capability Task**

- The functionality required to support the alpha version of the simulator is being developed at the Software Engineering Institute. The progress obtained to date indicates the system will meet its objective of demonstrating to stakeholders an alpha version of a simulator capability at the Software Engineering Institute as scheduled.
- Documentation of the simulation language, Easel, is progressing in a timely manner. The preliminary version of the Easel Language Reference Manual was completed in July 2000 and further refined in a follow-up draft in October 2000. The preliminary version of the Easel Author Guide is on schedule for delivery in January 2001.

<sup>5</sup> HIPAA is the Health Insurance Portability and Accountability Act of 1996; and NCQA is the National Committee for Quality Assurance.

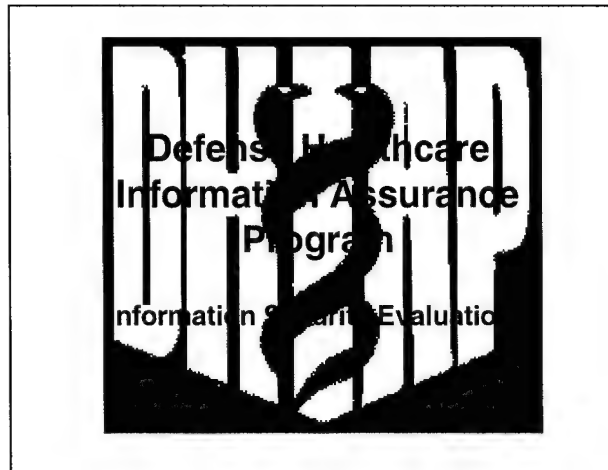
- The Easel simulation's use of a new approach to simulation, an emergent simulation language and environment employing a paradigm of property-based types (i.e., describing abstract classes of examples by their shared properties) is sufficiently robust to produce accurate conclusions about examples from the physical world.
- Formation of the Simulation Advisory Group (SAG) is important to fulfillment of all aspects of this task. It will be the SAG that helps focus development of the Easel, motivate its use in the MTF environment, and expand its usefulness as a generalizable tool. Recruitment of members for the SAG and formation of this group is work currently in progress.

### Bibliography

- [CER] Andrews, Archie, et al., *DHIAP Phase I Composite Evaluation Report*, ATI IPS Technical Report 00-02, February 2000.
- [IMG] Melton, Lane, *DHIAP Supplemental Installation and Maintenance Guide*, ATI IPT Special Report 00-03, April 2000.
- [TDR] Crane, Lynn, et al., *DHIAP Phase I Technology Demonstration Report: Prototype for Remote Authentication Dial-In User Service (RADIUS)*, ATI IPT Technical Report 00-04, April 2000.
- [RSA] Packard, Stephen, et al., *Remote System Administration: Issues and Recommendations*, ATI IPT Special Report 00-05, April 2000.
- [PKI] Streetman, Kibbee, et al., *Public Key Infrastructure: Resources, Requirements, and Recommendations*, ATI IPS Special Report 00-06, April 2000.
- [TRU] Stinson, Jack, et al., *Trust Model: Defining and Applying Generic Trust Relationships in a Networked Computing Environment*, ATI IPT Special Report 00-07, May 2000.
- [SIM] Fisher, David, *Survivability and Simulation*, personal correspondence of the author, August 2000.
- [RAD] Crane, Lynn, et al., *DHIAP RADIUS Implementation: Business Case Analysis*, ATI IPT Technical Report 00-08, October 2000.

**4. Appendices**

<b><u>APPENDIX</u></b>	<b><u>TITLE</u></b>	<b><u>PAGE</u></b>
A.	Army IMO Briefing .....	33
B.	RADIUS Supplemental Installation and Maintenance Guide .....	36
C-1.	Remote System Administration .....	105
C-2.	Public Key Infrastructure .....	117
C-3.	Trust Model .....	164
D.	Business Case Analysis Presentation .....	196
E.	Interim Program Review Presentation .....	200
F.	Octave Overview .....	202
G.	Site Executive Briefing .....	203
H.	Business Case Analysis #2 Scope .....	207
I.	Business Case Analysis #3 Scope .....	209
J.	Business Case Analysis #4 Scope .....	211
K.	Survivability and Simulation Description .....	213
L.	Easel Language Reference Manual Version 1.0.3.3 .....	218



Slide 1

### Overview of DHIAP

- > Originated from a Congressional mandate to
  - Evaluate the ability of the DoD medical healthcare system to protect patient information
  - Develop solutions and demonstrate system improvements
- > Awarded and managed by TATRC (Telemedicine and Advanced Technology Research Center)

### > DHIAP Goals

- Develop and apply tools and techniques that
  - Evaluate system and network risks
  - Address necessary doctrine, infrastructure, training, programmatic, and technology issues
  - Support Medical Information Security Readiness (MISR) strategy
  - Support Risk Information Management Resource (RIMR)
- Improve information assurance at Medical Treatment Facilities

Defense Healthcare Information Assurance Program

Slide 2

### The Team

MRMC - TATRC

	Advanced Technology Institute
	Software Engineering Institute CERT Coordination Center
	Lockheed Martin Energy Systems
	Arthur D. Little, Inc.
	Computerized Patient Records Institute Healthcare Open System & Trials
	KRM Associates

Defense Healthcare Information Assurance Program

Slide 3

### Areas of Concentration

- > Site Technical Assessments
- > Technology Prototype
- > Research Reports
  - Remote Administration
  - Public Key Infrastructure
  - Trust Model

Defense Healthcare Information Assurance Program

Slide 4

### Site Technical Assessments

Defense Healthcare Information Assurance Program

Slide 5

### Systems Examined

#### Technology Platforms

- Network infrastructure and world wide web
- Operating systems (UNIX, NT, Win95, VMS)

#### Application Systems

- CHCS - Composite Health Care System
- MDIS - Medical Diagnostic Imaging System
- CEIS - Corporate Executive Information System
- TPOCS - Third Party Outpatient Collection System
- ADS - Ambulatory Data System
- TAMMIS - Theater Army Medical Management Information System
- DBSS - Defense Blood Standard System
- MRS - Mammography Reporting System
- DMLSS - Defense Medical Logistics Standard System

Defense Healthcare Information Assurance Program

Slide 6

**ISE Observations****Vulnerability Categories**

- Organizational Climate
- Security Policy and Procedures
- Security of Patient Information
- Security Staffing
- External System Access
- System Administration
- Disaster Recovery and Backup
- Physical Site Security
- Security Training

• Specific Observations Included as Backup Slides

Defense Healthcare Information Assurance Program

Slide 7

**ISE Recommendations****Management Actions**

- Information protection oversight - a Command priority
- Organizational responsibility and authority for security
- Policy
- Procedures
- Technology standards
- Technology
- Training

• Specific Recommendations Included as Backup Slides

Defense Healthcare Information Assurance Program

Slide 8

**Summary of ISE Findings****Barriers**

Operational needs  
sometimes of  
policy declar

Training and  
procedures a  
not aligned  
with policy

Policy is imp  
rapidly evol  
technical capab  
and user expectations

**Solutions**

Understand risks and  
operational necessities

Deliver training and  
procedural guidance  
reinforce sound  
practices

Build on a solid base of  
policy that recognizes  
operational mandates and  
emerging capabilities

Defense Healthcare Information Assurance Program

Slide 9

**Summary of ISE Findings**

Defense Healthcare Information Assurance Program

Slide 10

**Summary of ISE Findings****Security Vision**

- Establish a level of tolerance
- Identify the organization's level of tolerance for risk

**Commitment**

- Assure Senior Management buy-in and support

**Training**

- Pay constant attention to system capability and security awareness

**Accountability**

- Clearly delineate who is doing what and who is responsible

**Security Architecture**

- Bridge the gap between policy and practices

Defense Healthcare Information Assurance Program

Slide 11

**ISE Recommendations****Recommended management emphasis:**

	Org. Climate	Security of Patient Info.	Security Policy & Proc.	Staff Support of Sec. Plan	External Systems & Apps.	System Admin.	Security Training	Disaster Recovery Backup	Physical Site Security
M									
A									
N									
A									
O									
G									
E									
M									
E									
P									
N									
T									
F									
O									
C									
U									
S									
A									
R									
E									
S									

Legend: ✓, ✓✓, and ✓✓✓ Indicate the Focus Area's relative impact on the Vulnerability Category

Specific recommendations for each level of command and for each of the nine observed categories provided in reports

Defense Healthcare Information Assurance Program

Slide 12



**ISE Recommendations****Create a strong security culture**

- **Establish the culture**
  - Identify subjects to be covered by security policy
  - Define the security model to be adapted and implemented
- **Enable the culture**
  - Change methods that impede protection of sensitive information
  - Provide resources (people, money, time, technology) necessary to define, implement, and monitor daily practice
  - Train the staff initially and continuously
- **Promote the culture**
  - Ensure compliance with security procedures and mandate meaningful penalties
  - Provide continued oversight of the information protection architecture and ensure it can address new technologies and operational practices

Defense Healthcare Information Assurance Program

Slide 13

**ISE Conclusions**

- Exploitable vulnerabilities exist
- Sound policy, implemented consistently and supported by the Chain of Command, is necessary and essential to information security
- Externally-mandated policy and programs impact a site's ability to secure sensitive information; coordination with external agencies is necessary
- Internal allocation of responsibilities impacts secure system configuration, management, and use
- Security-focused resources (training, staff time, priorities) must be allocated

Defense Healthcare Information Assurance Program

Slide 14

**Current Efforts**

- **Develop and apply tools and techniques that**
  - **Evaluate system and network risks**
  - **Address doctrine, infrastructure, training, programmatic, and technology issues**
  - **Support TATRC's Medical Information Security Readiness strategy and Risk Information and Management Resource**
- **Doctrine**
  - **Policy, Procedures and Practice Analysis**
    - Tri-service integrated team review of HIPAA and service regulations
- **Technology**
  - **Risk Information and Management Resource (RIMR)**
    - Repository of tools and information assets to support the IA requirements at the MTF
    - Self-Directed Risk Analysis tools
    - Technical Business Case Analysis
    - Database of lessons learned



Defense Healthcare Information Assurance Program

Slide 15

**Current Efforts (continued)**

- **Organization**
  - **Management Information Security Readiness (MISR)**
    - Security workshops in each Tricare region
    - Build MISR teams at each MTF
    - Support and sustain MISR training
      - Post workshops
      - RIMR



- **Bottom line - Improve information assurance for Army Medical Treatment Facilities**

Defense Healthcare Information Assurance Program

Slide 16

**For Additional Information**
<http://ips.atcorp.org/TR>
**Technical Reports**

- **Composite Evaluation Report**
  - Summary of Information Security Evaluations at MTFs, recommendations
- **Phase I Technology Demonstration Report - Prototype for Remote Authentication Dial-In User Service (RADIUS)**
  - Description of development of RADIUS-compliant technology for MTFs and recommendations for implementation
- **DHIAP RADIUS Supplemental Installation and Maintenance Guide**
  - Technical guide for installing and maintaining the DHIAP RADIUS prototype

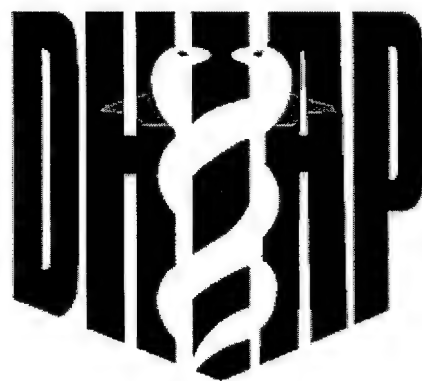
**Special Reports**

- **Remote Administration**
  - Summarizes issues and solutions for protection of information
- **Public Key Infrastructure**
  - Outlines requirements and use in the medical domain
- **Trust Model**
  - Examines the concept of a "trust model" in an integrated computer environment and its associated impact on risks

Defense Healthcare Information Assurance Program

Slide 17

# **DHIAP RADIUS Supplemental Installation and Maintenance Guide**



April 2000

ATI IPT Special Report 00-03

This work is supported by the U.S. Army Medical Research and Materiel Command under Contract No. DAMD 17-99-C-9001. The views, opinions and/or findings contained in this report are those of the authors and should not be construed as an official Department of the Army position, policy or decision unless so designated by other documentation.

# **DHIAP**

## **RADIUS Installation and Maintenance Guide**

---

### **PREFACE**

This report was prepared by the staff of the Information Protection Technology group of the Advanced Technology Institute as part of the Defense Healthcare Information Assurance Program (DHIAP). The DHIAP work is supported by the U.S. Army Medical Research and Materiel Command. As noted, the views, opinions and/or findings contained in this report are those of the authors and should not be construed as official Department of the Army position, policy or decision.

This Installation and Maintenance Guide is prepared as a supplement to existing Cisco and Microsoft instructions. It is written for the technically experienced system and network administrator based on lessons learned and experiences gained during the course of designing, installing, configuring, maintaining, and troubleshooting RADIUS compliant systems in the laboratory and at operational sites. It is written as a guide for installation and configuration of the Cisco and NT based systems that compose the RADIUS compliant system prototyped during the course of the DHIAP technology demonstration.

The specific guidance contained herein is drawn from many sources including official Cisco documentation, second-party guide books to Cisco and NT operations, extensive correspondence and discussion with Cisco technical representatives, and personal experiences.

Lane Melton was the principal author and overall coordinator for this guide. Assistance in clarifying and detailing technical approaches was provided by Dr. Jack Stinson, Mr. Forrest Schwengels of the Advanced Computing Technology staff of Lockheed Martin Energy Systems, and numerous members of the Cisco Technical Staff. Archie Andrews edited the report, and Stephen Pellissier proofread it. Sarah Hartline typed and revised the many drafts of the guide and prepared the report for publication.

Archie D. Andrews  
Principal Investigator, Defense Healthcare Information Assurance Program  
Director, Information Protection Technology  
Advanced Technology Institute

**DHIAP**  
**RADIUS Installation and Maintenance Guide**

---

**TABLE OF CONTENTS**

1	Introduction .....	42
1.1	Purpose.....	42
1.2	Overview .....	42
2	Router Configuration.....	42
2.1	Initial Router Setup .....	42
2.1.1	Network and Modem Module Installation .....	43
2.1.2	Powering Up the Router .....	43
2.2	Set-Up Utility.....	43
2.2.1	Card Settings .....	43
2.2.2	Analog Modem Modules/Interface Port Determination.....	45
2.2.3	IP Pool Setup.....	46
2.2.4	Router Administrative Configuration.....	48
2.3	Basic Router Commands and Modes .....	50
2.4	Router Configuration Script.....	51
2.5	Router Configuration/Access Lists .....	54
3	Windows NT Server .....	58
3.1	Installation Steps .....	58
3.2	Internet Information Server.....	58
3.3	Telnet .....	59
3.4	WINS/DNS .....	59
3.5	User Setup on the Primary Domain Controller (PDC) .....	59
4	CiscoSecure Software.....	62
4.1	Installing the CiscoSecure Software .....	62
4.2	Configuring CiscoSecure Software.....	69
4.2.1	Interface Configuration .....	71
4.2.2	System Configuration.....	72
4.2.3	Group Setup.....	75
4.2.4	User Setup .....	79
4.2.5	Network Configuration .....	84
5	Remote Client Configuration.....	87
5.1	Hardware Requirements.....	87
5.2	Software Requirements .....	87
5.3	Workstation Configuration .....	87
5.3.1	Windows 95/98 .....	87
5.3.2	Windows NT 4.0 .....	88
6	Backup/Restore Procedures .....	94
6.1	Backup .....	94
6.2	Restore .....	97
6.3	Log File Archive .....	98
6.3.1	Archive Schedule .....	99
6.3.2	Types of Log Files.....	99
6.3.3	TACACS+/VOIP Log Files .....	100
6.3.4	Archive and Deletion .....	100

**DHIAP**  
**RADIUS Installation and Maintenance Guide**

---

6.3.5	Debug Log Files Deletion .....	102
7	Uninterruptible Power Supply .....	103
8	Microsoft Exchange.....	103

# DHIAP

## RADIUS Installation and Maintenance Guide

---

### TABLE OF FIGURES

Figure 1 - Entering Configuration Mode.....	44
Figure 2 - Configure IP Pool .....	47
Figure 3 - IP Pool Configuration Results .....	47
Figure 4 - Create User Name.....	48
Figure 5 - Display of User Name .....	49
Figure 6 - NT User Setup Screen .....	59
Figure 7 - NT User Properties Screen .....	60
Figure 8 - NT User Dialin Information Screen .....	60
Figure 9 - CS Initial Screen.....	62
Figure 10 - CS Reminder Screen.....	63
Figure 11 - CS Authentication Database Configuration Screen .....	63
Figure 12 - CS Server Details Screen.....	64
Figure 13 - CS Advanced Options Screen .....	65
Figure 14 - CS Active Service Monitoring Screen .....	65
Figure 15 - CS NAS Configuration Screen.....	66
Figure 16 - CS Enable Secret Password Screen.....	67
Figure 17 - CS AAA Configuration Screen .....	68
Figure 18 - CS Service Initiation Screen.....	69
Figure 19 - CS GUI Interface Screen .....	70
Figure 20 - CS Interface Configuration Screen.....	71
Figure 21 - CS System Configuration Screen .....	72
Figure 22 - CS System Configuration (AAA Server IP Pools) Screen.....	74
Figure 23 - CS System Configuration (Submit) Screen.....	74
Figure 24 - CS Group Setup Screen .....	76
Figure 25 - CS Group Setup (Time-of-Day) Screen .....	77
Figure 26 - CS Group Setup (IP Assignment) Screen.....	77
Figure 27 - CS Group Setup (IETF RADIUS Attributes) Screen.....	78
Figure 28 - CS User Setup Screen.....	79
Figure 29 - CS User Setup (Passwords) Screen .....	81
Figure 30 - CS User Setup (Groups) Screen .....	81
Figure 31 - Distribution Table Screen.....	83
Figure 32 - Distribution Area Editing Screen .....	83
Figure 33 - CS Network Configuration Screen.....	84
Figure 34 - CS Network Configuration (Server Setup) Screen.....	85
Figure 35 - CS Network Configuration (AAA Servers) Screen.....	85
Figure 36 - CS Network Configuration (AAA Server Setup) Screen.....	86
Figure 37 - CS Network Configuration (Submit and Restart) Screen.....	86
Figure 38 - Initial Dial-Up Networking Screen .....	88
Figure 39 - Edit Phonebook Entry Screen.....	89
Figure 40 - Modem Configuration Screen .....	89
Figure 41 - Edit Phonebook Entry Screen – Server Tab .....	90
Figure 42 - PPP TCP/IP Settings Screen.....	90
Figure 43 - Edit Phonebook Entry Screen – Script Tab.....	91

# **DHIAP**

## **RADIUS Installation and Maintenance Guide**

---

Figure 44 - Edit Phonebook Entry Screen – Security Tab .....	91
Figure 45 - Edit Phonebook Entry Screen X.25 Tab.....	92
Figure 46 - Dial-Up Networking Screen .....	92
Figure 47 - User Preferences Screen .....	93
Figure 48 - Logon Preferences Screen .....	93
Figure 49 - Initial Backup Screen .....	95
Figure 50 - Drive Letter Backup Screen .....	95
Figure 51 - File Selection Backup Screen.....	96
Figure 52 - Backup Information Screen .....	96
Figure 53 - Initial Restore Screen .....	97
Figure 54 - Tape Selection Screen .....	97
Figure 55 - Restore Location Screen.....	98



# DHIAP

## RADIUS Installation and Maintenance Guide

---

## 1 INTRODUCTION

### 1.1 Purpose

The purpose of this instruction is to provide a ready reference to the CiscoSecure software and technical reference guides. It should be used as an aid for installation and operation of a typical Cisco router and Windows NT server for RADIUS compliance. These guides include "Cisco 3600 Hardware Installation Guide for Cisco 3600 Series and Cisco 2600 Series Routers", "Software Configuration Guide for Cisco 3600 Series and Cisco 2600 Series Routers", "Cisco Network Modules Hardware Installation Guide For Cisco 3600 and Cisco 2600 Series Routers", and "CiscoSecure 2.4 Users Guide". These manuals, included with the system, provide information not included in this text and should be considered the definitive source for configuration and maintenance guidance. The instructions in this text provide a synopsis of the installation and maintenance of the RADIUS compliant system. It also provides troubleshooting tips for specific incidents encountered during installation and operations.

### 1.2 Overview

Typically the RADIUS compliant system will be installed and configured at each site. Because each site is different, there will be minor configuration differences at all locations. System engineers will install and integrate the RADIUS compliant system with the sites' present network. The site System Administrators have the responsibility of adding and deleting users as well as mapping restrictions on all user accounts.

The following information is provided as a point of reference to the local site System Administrator. It will help explain what has been done during the initial setup process and aid in the maintenance of their new RADIUS compliant system.

The initial setup process will include:

- Router Setup and Configuration
- Configure Windows NT for use with CiscoSecure
- Load and Configure CiscoSecure
- Workstation Configuration
- Troubleshooting Tips

## 2 ROUTER CONFIGURATION

### 2.1 Initial Router Setup

The first task to be completed during the initial system installation is router setup and configuration. Before powering up the router, make sure that the FastEthernet Network Module and Modem Modules are installed.

# DHIAP

## RADIUS Installation and Maintenance Guide

---

### 2.1.1 Network and Modem Module Installation

To install the Network and Modem Modules, perform the following tasks:

1. Connect an Anti-Static wrist strap to the installer and the metal chassis of the router.
2. Remove the module cover plates from the back of the router.
3. Remove the modules from the packing bags.
4. Place the module card into the alignment grooves on the router and slide into the router.
5. Tighten the module screws hand tight.

### 2.1.2 Powering Up the Router

Make sure you use only the provided Cisco cable. Establish a connection between a computer or terminal and router. The cable should be plugged into the "con" port on the front panel of the router and serial port on the back of the computer.

1. Power up the computer and create a HyperTerminal session.
2. Power Up the router.

## 2.2 Set-Up Utility

When the router is powered up for the first time, it will need to be configured. A setup utility will automatically start and prompt the administrator for information. This utility is good for basic setup parameters. When using the utility, answer all prompted questions with site specific information. When the utility is complete, it will create a configuration file. Additional configuration parameters will need to be added at a later time to enable it to interact with the CiscoSecure software. The following sections list responses to the setup utility. This information can also be used when configuring the router to interact with the CiscoSecure software.

### 2.2.1 Card Settings

The router contains one FastEthernet card on slot and port 0/0. The card can be set for 10Mbps or 100 Mbps. It must be configured to match the configuration for each site. By default, it is set to 100Mbps. Changing the default configuration can be accomplished with the setup utility on the initial router power up or configured manually. To apply any configuration changes from outside the setup program, the administrator must be in the enable mode. This is done by typing "enable" and supplying the enable secret password when prompted to enter the enable mode. The password was established with the setup utility. To change the FastEthernet card speed manually, perform the following:

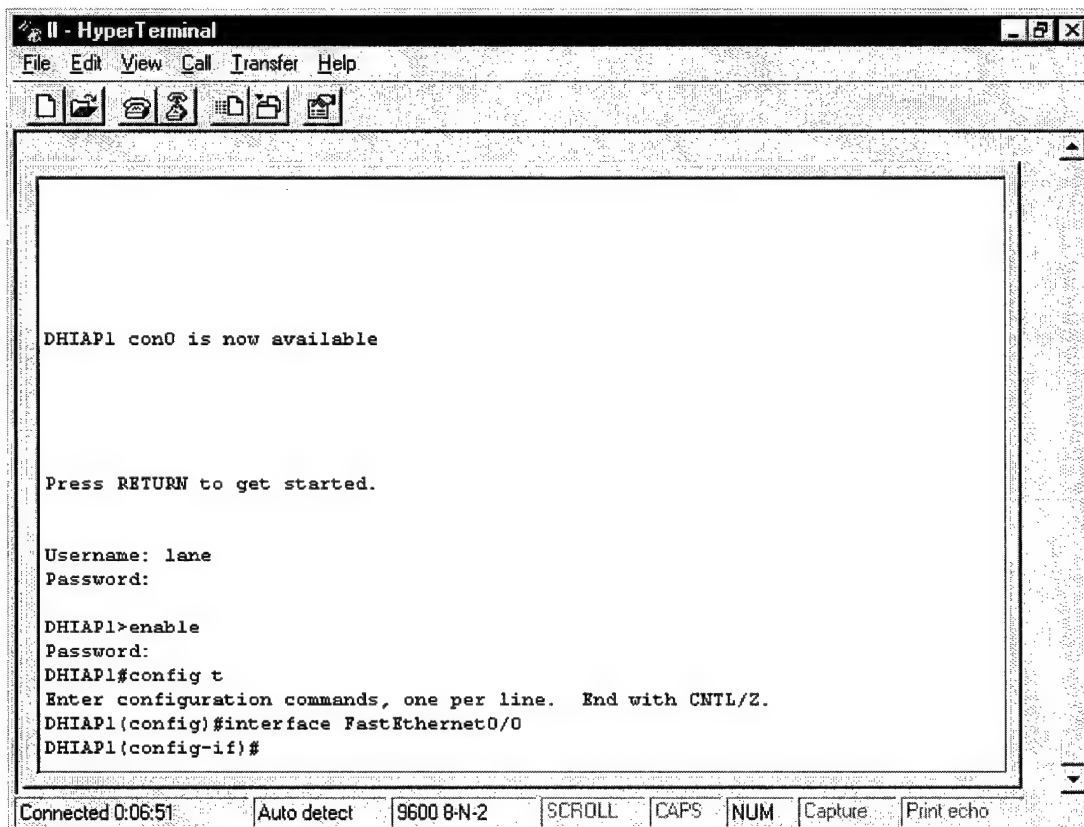
# DHIAP

## RADIUS Installation and Maintenance Guide

---

1. Enter router through HyperTerminal.
2. Log on as a user.
3. Go into enable mode. (This mode is encrypted and allows for configuration changes.)
4. Enter the enable password.
5. From the router # prompt, type *config t*.
6. This will put the administrator in the configuration mode.
7. The system prompt will display as router (config)#
8. Enter the following text:
9. *"interface fastethernet0/0"*
10. The system prompt will display as router (config-if)#
11. Enter the following text:
12. *"speed 10"* or *"speed 100"*
13. Enter *exit* to leave config-if mode and again to leave the config mode.

Note: The speed on the router must coincide with the speed of the network it is attached to.  
**Figure 1** displays the steps to enter the configure mode.



**Figure 1 - Entering Configuration Mode**

## DHIAP

### RADIUS Installation and Maintenance Guide

---

#### 2.2.2 Analog Modem Modules/Interface Port Determination

If the setup utility was used, the router should automatically recognize the modem modules and assign a port number to them. Should the router not recognize and assign numeric values to the modem ports, the administrator will have to do so. To calculate these port interface numbers, follow this procedure:

Interface-number = (32 x slot-number) + port number + 1  
(The slot number will be the router bay number in which the module resides.)  
On the Cisco 3640 there are four slots: slot 0, slot 1, slot 2, slot 3  
Bottom, right to left - slot 0, slot 1  
Top, right to left - slot 2, slot 3

Example:

Modem port 2/8 (slot 2 modem port 8)  
 $32 \times 2 = 64 + 8 = 72 + 1 = 73$   
Interface number for Modem Port 2/8 will be 73

It will be necessary to divide and place all modem ports into Interface Group-Async1 and Group-Async2. This procedure will accomplish the task of assigning port numbers to the available modem port slots. Once these groups are created, several configuration features also need to be added to them. It is possible that the router setup program has already numbered the modem ports. To verify this, type the following command:

***show mod*** or ***show modules*** or ***sh mod***

If no information appears, it will then be necessary to alter the configuration manually. Refer to the Cisco instruction manuals that accompanied the router to accomplish this task. Below is an example of the basic steps for manual configuration for the async-group.

1. From the router prompt, enter the enable mode by typing:  
***enable*** or ***en***
2. Enter the enable password and hit enter. A router #> will be displayed.
3. Enter the configuration mode by typing:  
***config t***
4. A router # (config) prompt will be displayed.
5. Type the following command to create Group-Async1:  
***interface group-async1***  
Hit the enter key  
Type the following commands, along with applicable information, and hit the enter key after each command. The explanation in parenthesis is for clarification only and should not have to be entered as part of the command.

## DHIAP

### RADIUS Installation and Maintenance Guide

---

<i>ip helper-address x.x.x.x</i>	(IP Address of the WINS server)
<i>ip directed-broadcast</i>	(If this feature is enabled, only those protocols configured using the IP forward-protocol global configuration command are forwarded).
<i>encapsulated ppp</i>	(enables PPP encapsulation)
<i>async mode interactive</i>	(To return a line that has been placed into dedicated asynchronous network mode to interactive mode, thereby enabling the slip and ppp EXEC commands)
<i>peer default ip address pool xyz</i>	(enter the name [replace xyz] of the IP Pool)
<i>no fair-queue</i>	(disable weighted fair queuing)
<i>no cdp enable</i>	(disable Cisco Discovery Protocol (CDP) on interface)
<i>ppp callback accept</i>	(allow callback)
<i>ppp authentication pap</i>	(use PAP authentication)
<i>group-range xx xx</i>	(enter the range of port numbers for the first group of analog modems, i.e., 65 80)

Type *exit* to leave the config mode

6. Verify that the group was created by typing:

*show interfaces*

7. The group should be displayed

The interface group should now be configured. Repeat the above steps to create a Group-Async2 for the second modem module ports.

#### 2.2.3 IP Pool Setup

The IP Pool for allocation of IP Addresses will normally be configured in the Graphical User Interface (GUI) of CiscoSecure (See Sec. 4). However, in the event the CiscoSecure Server is inoperative, the router must be configured to allocate IP Addresses in place of the CiscoSecure Server. An IP Pool must be set in the router's configuration file. To do this, the administrator must be logged in and in the enable mode. **Figure 2** displays the steps for creating the IP Pool, called "gen1". **Figure 3** displays the results of the configured IP Pool (note: **Figure 3** has IP Addresses masked, display will present local assigned IP Addresses).

# DHIAP

## RADIUS Installation and Maintenance Guide

```
II - HyperTerminal
File Edit View Call Transfer Help

Press RETURN to get started.

Username: lane
Password:

DHIAP1>enable
Password:
DHIAP1#config t
Enter configuration commands, one per line. End with CNTL/Z.
DHIAP1(config)#ip local pool gen1
DHIAP1(config)#exit
DHIAP1$

*Mar 1 00:44:24.119: %SYS-5-CONFIG_I: Configured from console by lane on consol
e

Connected 0:18:37  Auto detect  9600 8-N-2  SCROLL  CAPS  NUM  Capture  Print echo
```

Figure 2 - Configure IP Pool

```
II - HyperTerminal
File Edit View Call Transfer Help

interface Group-Async3
physical-layer async
no ip address
no ip directed-broadcast
!
interface Group-Async4
physical-layer async
no ip address
no ip directed-broadcast
!
ip local pool gen
ip local pool gen:
ip classless
no ip http server
!
access-list 150 permit tcp any host      eq 135
access-list 150 permit tcp any host      eq 139
access-list 150 permit udp any host      eq netbios-ns
access-list 150 permit udp any host      eq netbios-dgm
access-list 150 permit udp any host      eq netbios-ss
access-list 150 permit tcp any host      eq telnet
access-list 150 permit tcp any host      www
access-list 150 permit udp any host      eq domain
--More--

Connected 0:19:30  Auto detect  9600 8-N-2  SCROLL  CAPS  NUM  Capture  Print echo
```

Figure 3 - IP Pool Configuration Results

## DHIAP

### RADIUS Installation and Maintenance Guide

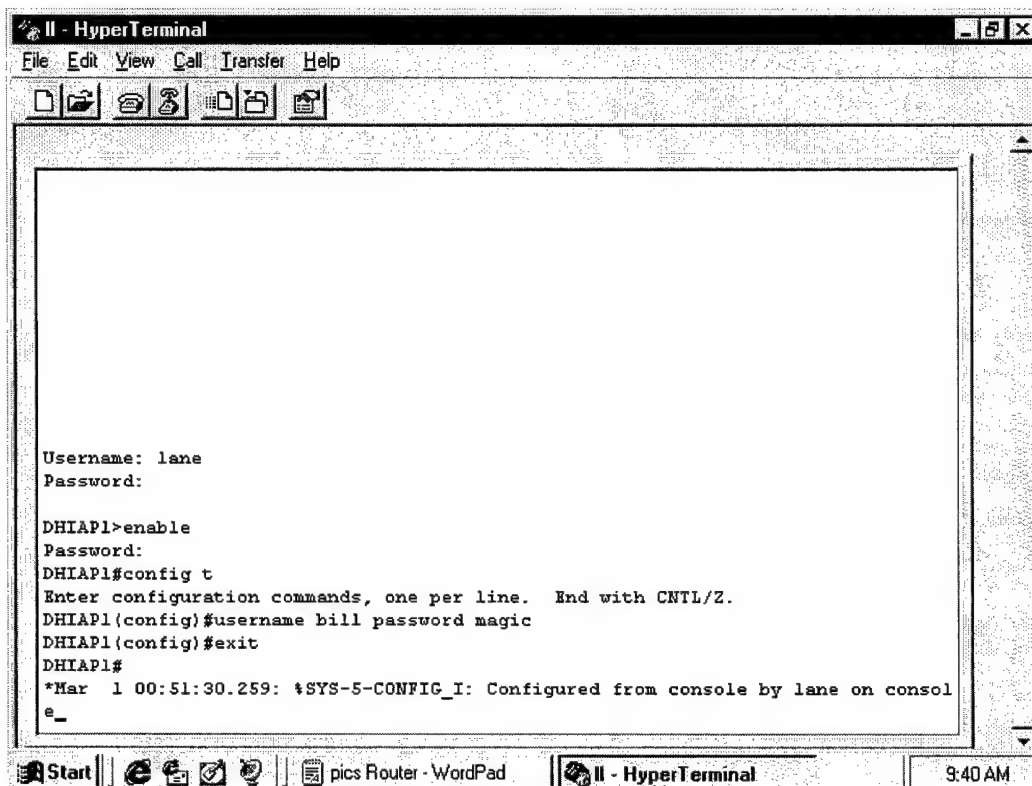
---

#### 2.2.4 Router Administrative Configuration

There must be at least one administrative user and password added to the router. Once the CiscoSecure software is loaded and configured, administrators will only be able to log into the router via a username and password created in the CiscoSecure database or Windows NT database. In the event these systems go down, administrators will need to be able to log into the router via the original username and password created during the initial configuration of the router. Setting an administrator password will allow this. If the setup utility was used for configuration, this step is not necessary. It can be used to add additional persons to the access list. To initially setup or input additional names manually, perform the following:

1. Enter the router through HyperTerminal.
2. Go into enable mode and enter the password.
3. Enter the following text.  
    *“username name password password”*.
4. Save the changes to permanent memory, NVRAM, by typing:  
    *“wri mem”*

**Figure 4** displays the procedures for entering the username and the result for the username of “bill” and a password of “magic.”



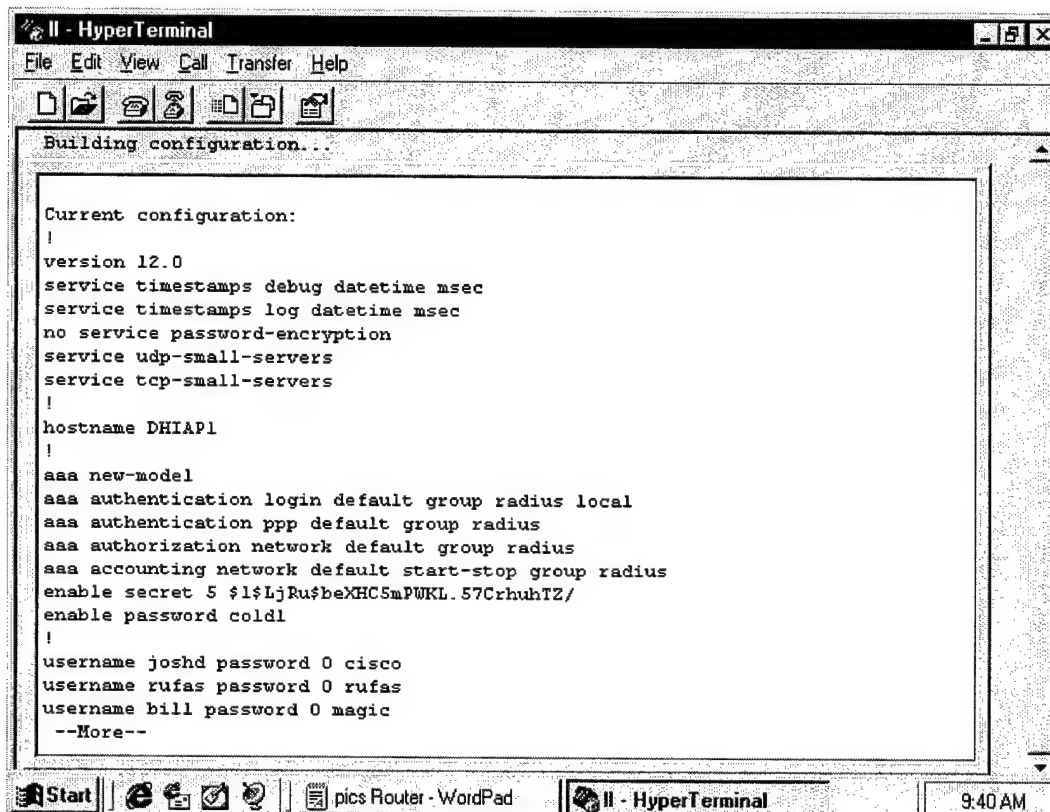
**Figure 4 - Create User Name**



# DHIAP

## RADIUS Installation and Maintenance Guide

Figure 5 displays the configuration file with the username and password entries.



```
Building configuration...

Current configuration:
!
version 12.0
service timestamps debug datetime msec
service timestamps log datetime msec
no service password-encryption
service udp-small-servers
service tcp-small-servers
!
hostname DHIAP1
!
aaa new-model
aaa authentication login default group radius local
aaa authentication ppp default group radius
aaa authorization network default group radius
aaa accounting network default start-stop group radius
enable secret 5 $1$LjRu$beXHC5mPWKL.57CrhuhTZ/
enable password coldl
!
username joshd password 0 cisco
username rufas password 0 rufas
username hill password 0 magic
--More--
```

Figure 5 - Display of User Name

**DHIAP**  
**RADIUS Installation and Maintenance Guide**

---

### 2.3 Basic Router Commands and Modes

Command or Mode *	Action
<i>enable</i>	Allow the router to be configured. Has an encrypted password.
<i>config t</i>	Allows the router to be configured. Places the router into the configure terminal mode.
<i>exit</i>	Will move the Administrator up a configuration level (exits the mode).
<i>debug</i>	Will provide debug information to the log and terminal pertaining to the string placed after it such as "debug aaa". This will provide information pertaining to CiscoSecure.
<i>debug RADIUS</i>	Debug option for RADIUS.
<i>debug callback</i>	Debug option for callback.
<i>debug aaa authentication</i>	Debug option for authentication.
<i>debug aaa authorization</i>	Debug option for authorization.
<i>debug aaa accounting</i>	Debug option for accounting.
<i>debug ppp neg</i>	Debug option for ppp negotiation.
<i>wri erase</i>	CAUTION: This will erase the router configuration from the NVRAM. It will erase everything.
<i>ctrl-z</i>	Will exit to a previous level.
<i>exit</i>	Will exit to a previous level.
<i>copy running-config startup config</i>	Will copy any changes made to the running configuration to the NVRAM for permanent use.
<i>no (setting)</i>	Used to undo any changes made to the configuration – "no speed 10" – will remove the 10 Mbs from the FastEthernet card speed.
<i>setup</i>	Used during initial setup. Should be avoided if the administrator has router experience.

\* All commands are terminated with a "return."

## DHIAP

### RADIUS Installation and Maintenance Guide

---

Mode	Usage	To Exit
exec	Mode entered during initial login to the router (enter).	logout
enable	Used to start configurations process. (enable)	disable, exit, ctrl-z
global configuration	Used to configure a terminal. (config t)	exit, end, ctrl-z
interface configuration	Used to configure an interface or a group. (interface fastethernet0/0)	exit, ctrl -z

#### 2.4 Router Configuration Script

The following is a recommended template provided as a basic router configuration model for a CiscoSecure RADIUS compliant system. It is configured for World Wide Web, Telnet and LAN use and is divided into sections. At the beginning of each section, there will be a brief explanation of the configuration options in underlined bold italics. Note that locally supplied information such as IP Addresses, user name, etc.; are bolded and in parentheses. They should be replaced with applicable values. Comments are in brackets [ ].

Router Config File  
Current configuration:  
!

**IOS Version, Date and Time Stamp, Use Current Time for Timestamp in Debug and Log. Do Not Apply Password Encryption as the Default.**

```
version 12.0
service timestamps debug datetime msec
service timestamps log datetime msec
no service password-encryption
service udp-small-servers
service tcp-small-servers
!
```

**Establish a Router Hostname, Establish CiscoSecure Process, Authentication, Authorization, Enable Tier Level Passwords**

```
hostname (HOSTNAME)
!
aaa new-model
aaa authentication login default group RADIUS local [Authenticate to CiscoSecure]
aaa authentication ppp default group RADIUS [Authenticate to CiscoSecure when dialing in]
aaa authorization network default group RADIUS [Assign group privileges to dial-in user]
aaa accounting network default start-stop group RADIUS [Turn on accounting]
enable secret (Secret Password) [Enable Secret and Enable modes require different passwords for varying configuration rights to the router.]
```

## DHIAP

### RADIUS Installation and Maintenance Guide

---

enable password (**Password**)

!

#### **Establish Administrative User Names**

username (**User ID**) password 0 (**Password**) [Establish User ID/Password for router administrators.]

username (**User ID**) password 0 (**Password**)

!

!

#### **Establish Domain-Name Server of Connecting Network, Establish IP Address of that Server**

ip subnet-zero

ip domain-name (**Domain Name**)

ip name-server (**DNS Server Name**)

!

#### **Establish Virtual Profile of the Usable Network, Establish Script for Callback Service, Establish Profile for CiscoSecure**

virtual-profile virtual-template 1

virtual-profile aaa

async-bootp nbns-server (**IP Address of WINS Server**)

chat-script offhook "" "ATH1" OK

chat-script callback ABORT ERROR ABORT BUSY "" "ATZ" OK "ATDT\T"

TIMEOUT 30 CONN

ECT \c

!

!

#### **Establish Process Time of 200 Seconds**

process-max-time 200

!

#### **Establish FastEthernet Card IP Address, Helper Address of Adjoining Network, Establish Directed Broadcast for Network Login, Establish FastEthernet Card Speed of 10 Mbps or 100Mbps**

interface FastEthernet0/0

ip address (**IP Address of FastEthernet0/0**) (**IP Address of subnet mask to default gateway**)

ip helper-address (**IP Address WINS Server**)

ip directed-broadcast

speed (**10 or 100**) [Depending on the line speed at each site]

!

#### **Apply Virtual Template to FastEthernet, Establish An IP Pool Name, Establish PPP Callback and PPP Name**

interface Virtual-Template1

ip unnumbered FastEthernet0/0

ip directed-broadcast

peer default ip address pool (**Name of IP Pool**)

ppp callback accept

ppp authentication pap

!

## DHIAP

### RADIUS Installation and Maintenance Guide

---

#### **Establish Group-Async For Modem Ports and Applied Router Variables to that Group**

```
interface Group-Async1
ip unnumbered FastEthernet0/0
ip helper-address (IP Address of WINS Server)
ip directed-broadcast [Accept broadcast]
encapsulation ppp [Accept encapsulated ppp]
async mode interactive [Enable interactive mode for async group]
peer default ip address pool gen
ppp callback accept [Allow callback]
ppp authentication pap [Allow standard user authentication]
group-range 65 80
!
```

#### **Establish Group-Async For Modem Ports and Applied Router Variables to that Group**

```
interface Group-Async2
ip unnumbered FastEthernet0/0
ip helper-address (IP Address of WINS Server)
ip directed-broadcast
encapsulation ppp
async mode interactive
peer default ip address pool gen
ppp callback accept
ppp authentication pap
group-range 33 40
!
```

#### **Establish IP Pool In Case CiscoSecure Goes Down**

```
ip local pool gen (IP Address range of local pool – X.X.X.X Y.Y.Y.Y )
ip route 0.0.0.0 0.0.0.0 (IP Address Of Subnet Mask For Default Gateway) [Route
all traffic from this address, 0.0.0.0, and subnet mask, 0.0.0.0., to IP Address of subnet mask for
default gateway]
no ip http server
!
```

#### **Establish Access List for CiscoSecure Filter, Dialer Protocol, Establish RADIUS Use on NAS and Key Between NAS and ACS**

```
access-list 150 permit tcp any host (IP Address for LAN Login) eq 135 [Allows
LAN Logon]
access-list 150 permit tcp any host (IP Address for LAN Login) eq 139 [Allows
LAN Logon]
access-list 150 permit udp any host (IP Address of WINS Server) eq netbios-
ns [Allows LAN Logon]
access-list 150 permit udp any host (IP Address of for sending datagrams) eq
netbios-dgm [Allows usage of datagram packets]
access-list 150 permit tcp any host (IP Address of telnet location) eq telnet
[Allows telnet]
access-list 150 permit tcp any host (IP Address of web location) eq www
[Allows link to WWW address]
```

## DHIAP

### RADIUS Installation and Maintenance Guide

---

```
access-list 150 permit tcp any host (IP Address of web location) eq www[Allows
link to WWW address]
access-list 150 permit udp any host (IP Address of DNS Server) eq domain
[Identifies DNS Server for authentication]
access-list 150 deny ip any any [Deny any IP Addresses, protocols or ports other
than those listed above, last line of Access Control List]
RADIUS-server configure-nas [RADIUS boot uses vender specs for CiscoSecure]
RADIUS-server host (IP Address of RADIUS Server) auth-port 1645 acct-port
1646
RADIUS-server key [Secret Key established at installation. Administrator choice during
installation]
!
```

#### **Establish Modem Callback, Modem Ports, PPP Connects, Accept Incoming and Outgoing calls, Various Passwords**

```
line con 0 [Allows con port connection]
transport input none
line 33 40 [Activates modem ports for Group-Async2]
autoselect during-login
autoselect ppp [Enable ppp connection]
script modem-off-hook offhook [Modem script to activate modem]
script callback callback [Allows modem callback]
modem InOut [Allows in and outbound calls]
transport input all
line 65 80 [Activates modem ports for Group-Async1]
autoselect during-login
autoselect ppp
script modem-off-hook offhook
script callback callback
modem InOut
transport input all
line aux 0 [Allows auxiliary connection]
line vty 0 4 [Allows terminal connection]
password summer2
!
!
end
```

## **2.5 Router Configuration/Access Lists**

At this point, the router has been configured for standard operation and supports one network module, one sixteen port modem module and one eight port modem module. Configuration parameters also include support for site LAN login, as well as World Wide Web use. Network utilities such as Ping and Telnet may also be utilized.

When the router is initially configured, it will allow full and complete transmission of all data. One objective of the RADIUS compliant system is to restrict certain types of data flow. Therefore, rights and restrictions are imposed on dialed in users. These rights and restrictions

## DHIAP

### RADIUS Installation and Maintenance Guide

---

are governed via an access list. When an access list is imposed, an implied "deny everything" is implemented. This will block out all data transmission between the router and the CiscoSecure Server. Data that will flow between the two is for authentication purposes only.

An access list must be created to specifically allow types of data flow to pass through the router. An access list will also govern where data can be passed to and from. For instance, telnet data can be prohibited while complete World Wide Web browsing data can be passed. Limited web browsing can be implemented to one site or none at all.

Access to the local area network can be permitted or denied. However, if permitted, the remote user will have no more or less privileges than when logged on locally.

Various access lists can be created on the router and then implemented or utilized by the CiscoSecure ACS Software. This process is called filtering. A filter is implemented through the CiscoSecure GUI for each group using an Access Control List (ACL) located on the router. An ACL will deny all traffic types to all locations by default. Only the traffic type and locations specified are permitted to pass through the router.

The example ACL labeled Access List "A," found below, will give users the ability to dial in remotely, Telnet to the CHCS system, and use the World Wide Web to browse to one MS-Exchange Mail site. All other activities are prohibited.

#### Access List "A"

```
access-list 160 permit tcp any host (IP Address for Primary CHCS) eq telnet
access-list 160 permit tcp any host (IP Address for Secondary CHCS) eq telnet
access-list 160 permit tcp any host (IP Address destination web address, i.e., Exchange
Server allows SSL passage) eq 443
access-list 160 permit tcp any host (IP Address destination web address, i.e., Exchange
Server) eq www
access-list 160 permit udp any host (IP Address of primary Domain Server) eq domain
access-list 160 deny ip any any [Will deny any traffic from this point onward. Anything
entered after this statement will not be included in the search.]
```

The example ACL labeled Access List "B" will give users the ability to dial-in remotely, Telnet to the CHCS system, check MS-Exchange e-mail via the WWW and log onto the site's Local Area Network (LAN). Other entries within the ACL support these actions.

#### Access List "B"

```
access-list 150 permit tcp any host (IP Address of PDC) eq 139
access-list 150 permit udp any host (IP Address of WINS) eq netbios-ns
access-list 150 permit udp any host (IP Address of PDC to pass datagrams) eq netbios-dgm
access-list 150 permit tcp any host (IP Address for CHCS) eq telnet
access-list 150 permit tcp any host (IP Address destination web address, i.e., Exchange
Server) eq www
```



## DHIAP

### RADIUS Installation and Maintenance Guide

---

```
access-list 150 permit tcp any host (IP Address destination web address, i.e., Exchange
Server allows SSL passage) eq 443
access-list 150 permit udp any host (IP Address of primary Domain Server) eq domain
access-list 150 permit udp any host (IP Address of secondary Domain Server) eq domain
access-list 150 deny ip any any [Will deny any traffic from this point onward. Anything
entered after this statement will not be included in the search.]
```

Note: As pointed out in Section 2.4, the parentheses in the commands listed above are not a part of the syntax that must be entered. The syntax entry should appear as follows:

```
access-list 150 permit tcp any host X.X.X.X eq 139
```

Many ACLs can exist on the router. They are classified by number, i.e., 150, 160, etc., and implemented through the CiscoSecure administrative software. For instance, general users could have ACL 160 restricting their access while network administrators could have ACL 150 restricting their access. The restrictions could also be imposed on general users and not system administrators. The network administrators would have full and complete access to the local area network, World Wide Web and telnet privileges. General users would be limited.

The above access lists define exactly what types of traffic can pass through a router and to what destination those packets can travel. Each list is defined with a numeric valued name. For TCP traffic the name range is from 99 to 199. In the event that two Medical Treatment Facilities (MTFs) are using the same Access Control Lists (ACLs), the lists are given two different titles, for instance 150 and 160. It is recommended that for future development, each site be given a range of numbers for access list assignment. This will eliminate any confusion of rights assignment should the sites ever be combined. The list will define whether a specific traffic is permitted or denied, i.e., "permit TCP (or UDP)". In the present access list the traffic is permitted from any host to a specified IP Address. The "eq" defines the port assigned to the traffic sent. It will state if it is a telnet or web transmission. It can also state if it will be a request for Secure Socket Layers or a domain request. The access list is not limited to the above example. It can be modified to accommodate the site's needs. The more liberal access that is given, the less secure a site is.

#### **Troubleshooting Tip: Administrative Shutdown**

**Problem:** An administrative shutdown is a software command activated in the router's configuration file that removes from operation various components of the router. Its purpose is to prevent permanent system damage and to alert the system administrator of a potential problem. The router, not the system administrator, performs an administrative shutdown automatically. A corrupt router configuration file, module failure, router component failure or a power fluctuation can cause an administrative shutdown.

Two instances have occurred: A power "spike" and corruption of the NAS configuration file.

**Instance One:** The router, connected to an uninterruptible power supply (UPS), experienced a mild power spike. While the UPS prevented internal damage to the router, an administrative shutdown was induced. During testing, users were no longer able to authenticate using the CiscoSecure system. Various physical layer connection tests

## DHIAP

### RADIUS Installation and Maintenance Guide

---

were performed on the router and all were successful; however, there was no transfer of data from the router to the server. The router's configuration file was reviewed and the administrative shutdown was noted in the FastEthernet0/0 section. It appeared as "shutdown".

**Instance Two:** The second instance of the administrative shutdown occurred when changes were made to the router configuration file. The changes corrupted the file and induced an administrative shutdown in the FastEthernet0/0 section.

**Symptoms:**

1. Client timing out while being authenticated. Usually this occurred on a Windows NT client.
2. Message reading that the system was unable to authenticate due to a bad password or user ID.
3. The administrative icon on the AAA Server desktop will only activate a cached copy of the CiscoSecure administrator.
4. NAS and AAA Server are no longer able to communicate.
5. Unable to telnet into the NAS from the AAA Server
6. Unable to telnet into the NAS via another device on the local network.
7. The administrative icon on the AAA Server desktop will not activate the administrator page. It will return an error stating that it was unable to establish a connection with the address of the NAS.

**Solution:** The administrative shutdown will usually appear in the FastEthernet0/0 section of the configuration file as "shutdown". Locate this command and remove it. Recompile the configuration file and write the changes to memory and test for proper operation. Detailed steps to complete this process are listed below.

**Detailed correction steps:**

1. Stop all services relating to CiscoSecure in the Control Panel, Services Section of Windows NT.
2. Establish a "Con" session by attaching the cable provided into the "Con" port on the front of the NAS and the AAA Server.
3. Establish a HyperTerminal session with the NAS and log in as a user that was established in the original configuration file.
4. From the prompt type, "Show IP Interface Brief" or "show int FastEthernet 0/0".
5. It will give operational information concerning the interface. It will state that the interface is up or down and that the protocol is up or down. In the event of a shutdown, the interface and protocol should read "down".
6. Enter the enable mode by typing "enable" and supply the appropriate password.
7. Type "show config" to display the configuration file. Scroll down to the FastEthernet 0/0 section and confirm the presence of the "shutdown" command.
8. Enter the configuration mode by typing "config t".
9. Type "interface fastethernet0/0" and hit enter to enter the configuration section of the FastEthernet Card.
10. Type "no shutdown" and hit ctrl-z. Type "wri mem" to make the configuration changes permanent.
11. Type "show config" to confirm the deletion of the "shutdown" statement.

## **DHIAP**

### **RADIUS Installation and Maintenance Guide**

---

### **3 WINDOWS NT SERVER**

The Windows NT operating system is the required platform for the CiscoSecure software. It must be loaded and configured prior to the installation of the CiscoSecure server software. Before loading the NT Server software, establish the machine name and IP Address of the server, default gateways, DNS server, Primary Domain Controller (PDC) and WINS server.

Several software packages must be installed with the Windows NT. They are MS-Internet Information Server, MS-Service Pack 4 and MS-Options Pack 4.0. Select to upgrade the browser when prompted by MS-Options Pack 4.0. This software configuration has been installed on all present systems and extensively tested. When installing the software, defaults can be used on all options.

During the initial installation of Windows NT software, add the server to the network domain. This will require the use of an account with administrative privileges and the ability to add computers to the domain. Once the installation is complete, test the system for log-in ability and appropriate use of network neighborhood, Internet capability, telnet function and standard communications between various equipment.

#### **3.1 Installation Steps**

Check all hardware for proper installation. Make sure that the mouse, keyboard, monitor and all backup drives are installed into the server before the installation.

1. Start the Win NT installation process.
2. Let the system detect all new hardware.
3. Accept default settings.
4. Load Win NT on C:\.
5. If asked to reformat, do so.
6. Set NTFS for the file system.
7. Name the Server. (This name will be needed again during the CiscoSecure loading process.)
8. Select stand-alone server.
9. When prompted to setup networking, do so. (Recommend using the drivers that came with the NIC Card as opposed to letting Win NT find the drivers.)
10. Set the card to 10 or 100Mbps and proper mode.
11. Upgrade the browser and all Y2K Patches.
12. Ensure that MS-Internet Information Server is installed.
13. Install MS-Service Pack 4. (Disregard all messages that would indicate that Service Pack will not work or has not been tested with current software.)

#### **3.2 Internet Information Server**

Install this software and accept all default settings.

# DHIAP

## RADIUS Installation and Maintenance Guide

### 3.3 Telnet

There could be instances that the telnet function will not work after a standard Win NT installation. Test this feature before loading the CiscoSecure Software. If it is not loaded, install a telnet daemon.

### 3.4 WINS/DNS

Supply the appropriate WINS and DNS IP Addresses.

### 3.5 User Setup on the Primary Domain Controller (PDC)

User IDs and Passwords located on the PDC can be used by the CiscoSecure system to authenticate dial-in users. Users that provide their User ID and Password from the user database on the PDC must be configured to do so. Using the Windows User Manager for Domains, configure all user rights as required by the LAN and customary access. Ensure that the user has dial-in permission. **Figure 6** displays the User Manager for Domains screen. To configure each user for dial-in permissions perform the following steps:

1. From the User Manager, double click a user or create a new user.

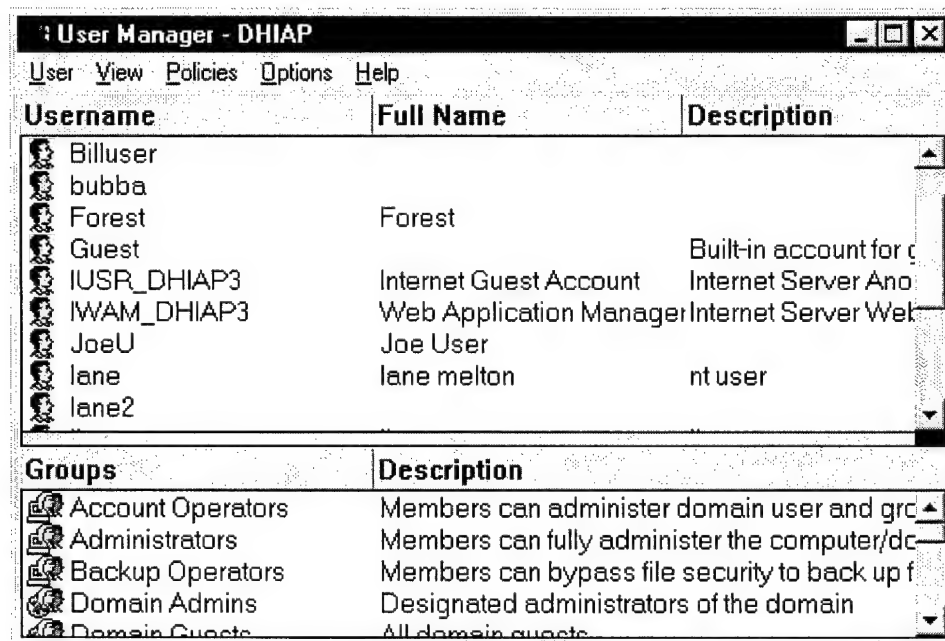


Figure 6 - NT User Setup Screen

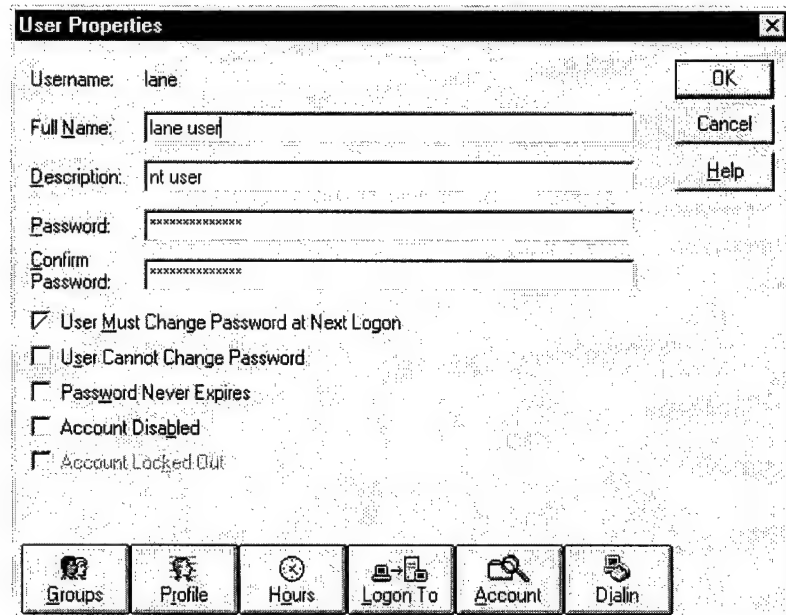
# DHIAP

## RADIUS Installation and Maintenance Guide

---

**Figure 7** displays the User Properties screen.

2. Click the Dialin option at the bottom of the option box.

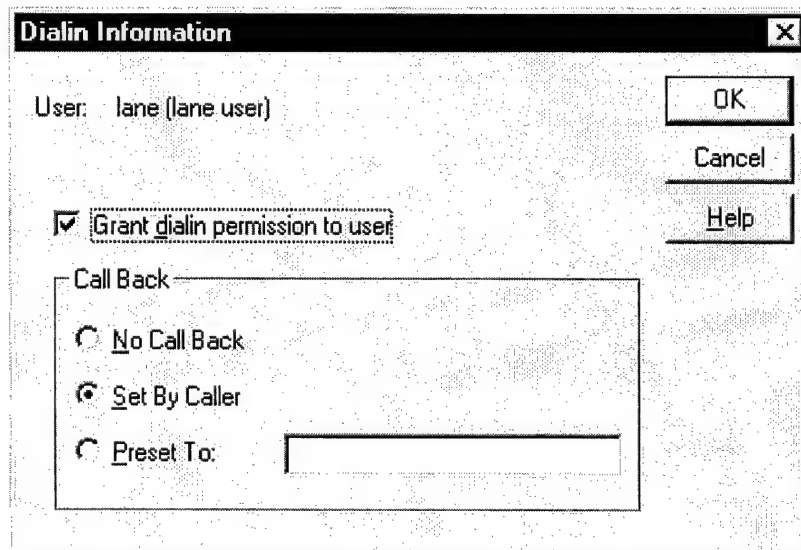


The 'User Properties' dialog box is shown. It has a title bar with a close button. The fields are: Username: lane, Full Name: lane user, Description: nt user, Password: (masked with asterisks), and Confirm Password: (masked with asterisks). To the right are buttons for OK, Cancel, and Help. Below the fields are five checkboxes: 'User Must Change Password at Next Logon' (checked), 'User Cannot Change Password' (unchecked), 'Password Never Expires' (unchecked), 'Account Disabled' (unchecked), and 'Account Locked Out' (unchecked). At the bottom is a row of six icons with labels: Groups, Profile, Hours, Logon To, Account, and Dialin.

**Figure 7 - NT User Properties Screen**

**Figure 8** displays the Dialin Information screen.

3. Click the check box "Grant dialin permission to user."
4. Select the radio button "Set By Caller"
5. Click OK



The 'Dialin Information' dialog box is shown. It has a title bar with a close button. The 'User' field shows 'lane (lane user)'. To the right are buttons for OK, Cancel, and Help. Below the user field is a checked checkbox labeled 'Grant dialin permission to user'. Underneath is a 'Call Back' section with three radio buttons: 'No Call Back' (unchecked), 'Set By Caller' (checked), and 'Preset To:' (unchecked) followed by an empty text box.

**Figure 8 - NT User Dialin Information Screen**

## DHIAP

### RADIUS Installation and Maintenance Guide

---

#### **Troubleshooting Tip: Access to Network Drives**

##### **Problem:**

CiscoSecure database members, not Windows NT database members, have access to non-restricted, shared network drives on the LAN. This occurs when no ACL has been implemented.

##### **Solution:**

Access restrictions must be set for all network drives on the LAN. Apply an ACL to all dial-in users.

##### **Analysis:**

System administrators should be aware that this condition exists and should take corrective action to ensure that all areas of the LAN have access restrictions imposed on them.

#### **Troubleshooting Tip: Dial-in Users are Denied Access To The Internet**

##### **Problem:**

All dial-in users have authorization permissions set by an access list defined within the Network Access Server, the router. The access list will allow the dial-in user to gain access to specified Internet sites. On occasion users with these privileges will not be able to access it. When the dial-in user's browser is activated, the site name is resolved to an IP Address, but the site never displays on the screen; the browser times out.

This timeout is due to the lack of a clearly defined path from the present network to the Internet gateway. To determine if a clear path is defined, go to the command prompt at the dial-in client and type "ipconfig". Two types of information should be displayed. The first will be the local LAN adapter information and should be correct for connecting to the LAN via a local onsite RJ45 connection. The second will be of the PPP Adapter. The following information should be listed:

IP Address – (IP assigned from CiscoSecure Router Pool)  
Subnet Mask – (Mask of the subnet that leads to default gateway)  
Default Gateway – (Router to the outside world or LAN)

##### **Solution:**

If the information pertaining to the PPP Adapter is not correct, the appropriate changes need to be made to the Network Access Server configuration file. Check for the correct information on the following lines:

```
ip domain-name (domain name)
ip name-server (DNS Server)

Interface FastEthernet0/0
ip address (module_address subnet_mask)

ip route 0.0.0.0 0.0.0.0 (external gateway IP Address)
```

##### **Analysis:**

If the dial-in user is unable to "bring up" an Internet site, the path to the Internet is not sufficiently mapped within the router. When changes to the path are implemented, a successful site connection will result.

## DHIAP

### RADIUS Installation and Maintenance Guide

---

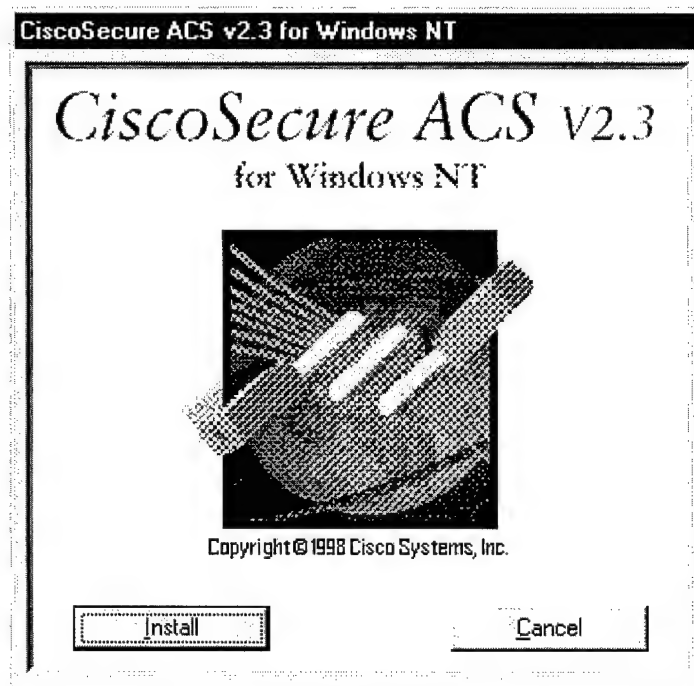
After the Windows NT installation is complete and before loading the CiscoSecure Software, do the following:

1. Grant Dial-in Permissions to users via Win NT.
2. Make sure that Win NT can ping the NAS.
3. Update browsers (MSIE – 3.02 or higher or Netscape 3.x or Communicator 4.x or higher).
4. Enable Java support.

## 4 CISCOSECURE SOFTWARE

### 4.1 Installing the CiscoSecure Software

After a complete Windows NT installation, insert the CiscoSecure CD-ROM into the server and start the installation process. When the program is activated, the following screen will appear:



**Figure 9 - CS Initial Screen**

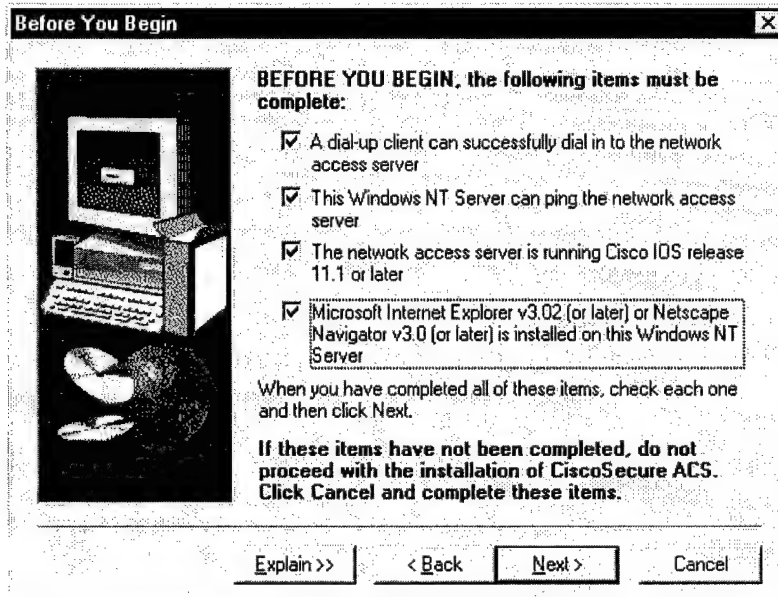


## DHIAP

### RADIUS Installation and Maintenance Guide

---

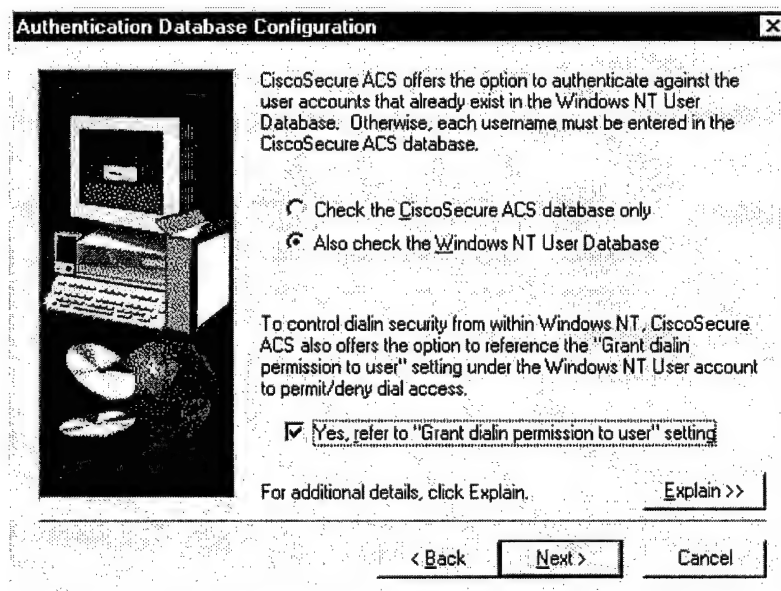
**Figure 10** displays the screen to remind the installer of the preinstall checks to be performed. Accept the defaults for network communication components listed.



**Figure 10 - CS Reminder Screen**

When loading the CiscoSecure Software (see **Figure 11**):

1. If the RADIUS compliant system will be used in conjunction with Win NT Databases, click the radio button that selects CiscoSecure and Win NT.
2. If using the system with Win NT, click the check box "Grant dialin permission to user."
3. Click Next.



**Figure 11 - CS Authentication Database Configuration Screen**

## DHIAP

### RADIUS Installation and Maintenance Guide

---

During the CiscoSecure installation process, the router will take on the responsibility of a Network Access Server (NAS). From this point forward, the router will be referred to as a NAS. When configuring the NAS, the following fields should be filled in according to site specific information (see **Figure 12**):

1. Authenticate Users Using: Click the drop down box and choose – RADIUS (Cisco) and hit the tab key.
2. Access Server Name: Enter the appropriate name for the access server. This will be the router's name.
3. Access Server IP Address: Enter the IP Address of the Access Server (i.e., the router).
4. Windows NT Server IP Address: If networking has already been established and is operational, the IP Address will be in this location.
5. TACACS+ or RADIUS Key: This is a secret key which the router uses to communicate with the RADIUS Server. It should be alphanumeric and at least eight characters long.
6. After entering the correct information, click Next.
7. The software will start loading.

**CiscoSecure ACS Network Access Server Details**

To successfully configure CiscoSecure ACS to communicate with your first NAS, the following information is required. Additional NASes can be configured from within CiscoSecure ACS once installed.

Authenticate Users Using: RADIUS (Cisco)

Access Server Name: DHIAP1

Access Server IP Address:

Windows NT Server IP Address:

TACACS+ or RADIUS Key:

[Explain >>](#)

< Back Next > Cancel

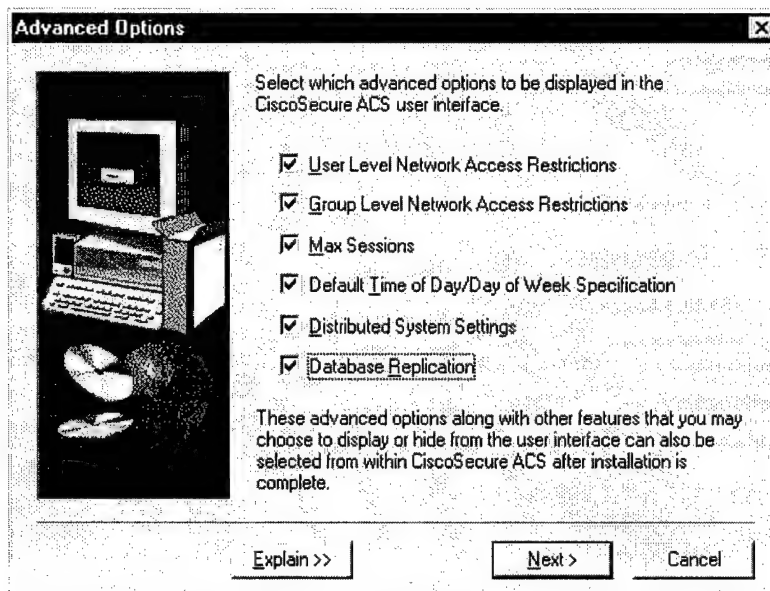
**Figure 12 - CS Server Details Screen**

## DHIAP

### RADIUS Installation and Maintenance Guide

---

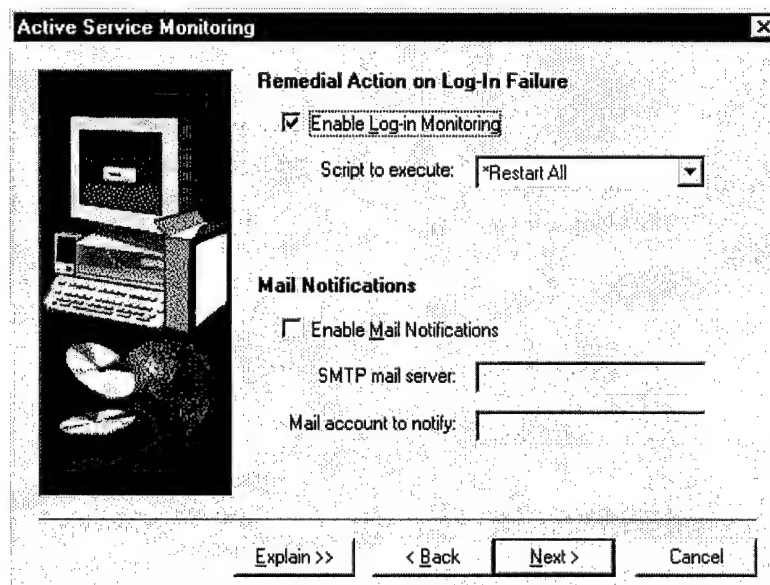
An Advanced Options screen is displayed in **Figure 13**. Select all check boxes and click Next.



**Figure 13 - CS Advanced Options Screen**

An Active Service Monitoring screen is displayed in **Figure 14**. Click:

1. Enable Log-in Monitoring.
2. From the "Script to execute" option, select:  
    \*Restart All
3. Do not configure mail notification at this time.
4. Click Next.



**Figure 14 - CS Active Service Monitoring Screen**

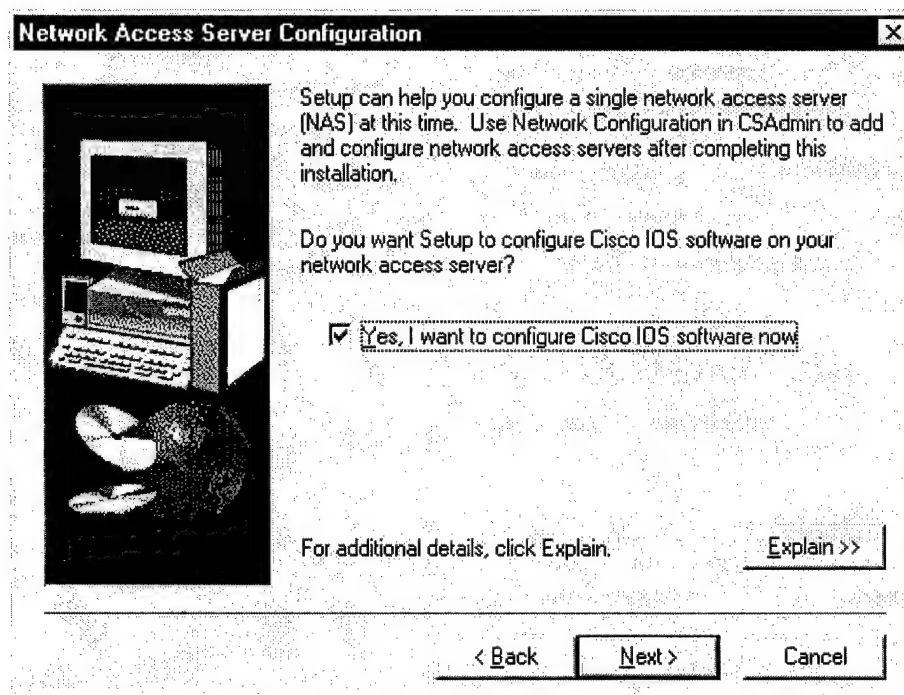
## DHIAP

### RADIUS Installation and Maintenance Guide

---

A Network Access Server Configuration screen is displayed in **Figure 15**.

1. Click the check box “Yes, I want to configure Cisco IOS software now” if this is a new installation.
2. Click Next.



**Figure 15 - CS NAS Configuration Screen**

**IMPORTANT NOTE** – This can be used for the first CiscoSecure Configuration of the NAS (router). However, should a second AAA Server need to be configured, bypass this screen and configure the NAS manually. Do NOT check the box, “Yes, I want to configure Cisco IOS software now.” Enter the additional information in the global configuration mode within the router. Proceed to the CiscoSecure ACS Service Initiation Screen Section.

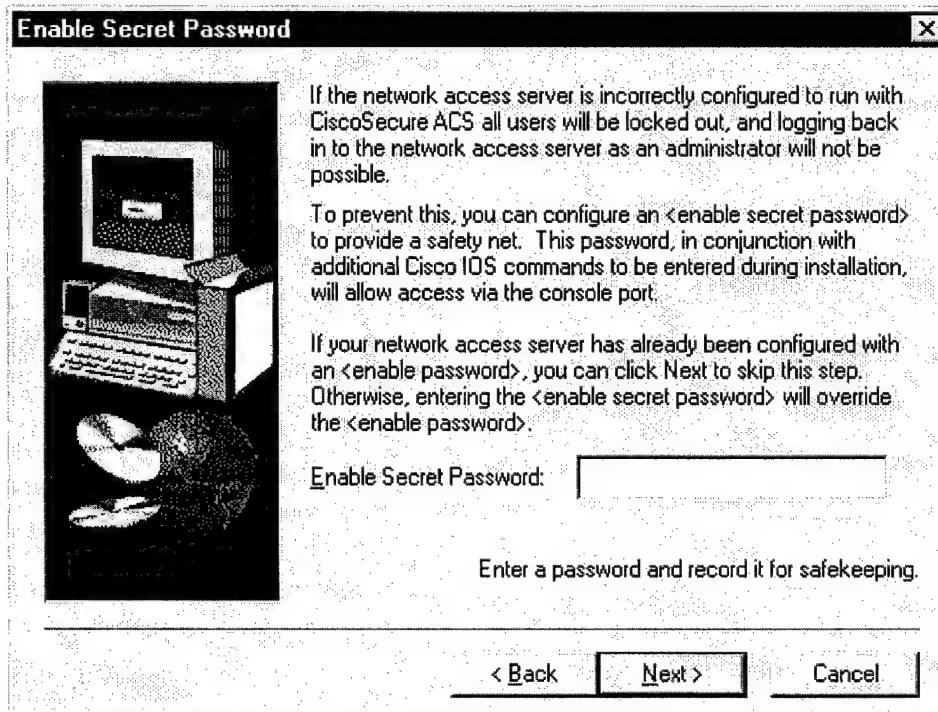
## DHIAP

### RADIUS Installation and Maintenance Guide

---

An Enable Secret Password screen is displayed in **Figure 16**. Selecting this option will enable the NAS (router) and the CiscoSecure software to interact so that the configuration process can be completed. If the enable secret password has already been established at the NAS, click Next and skip this step.

1. Enter the Enable Secret Password.
2. Click Next.



**Figure 16 - CS Enable Secret Password Screen**

## DHIAP

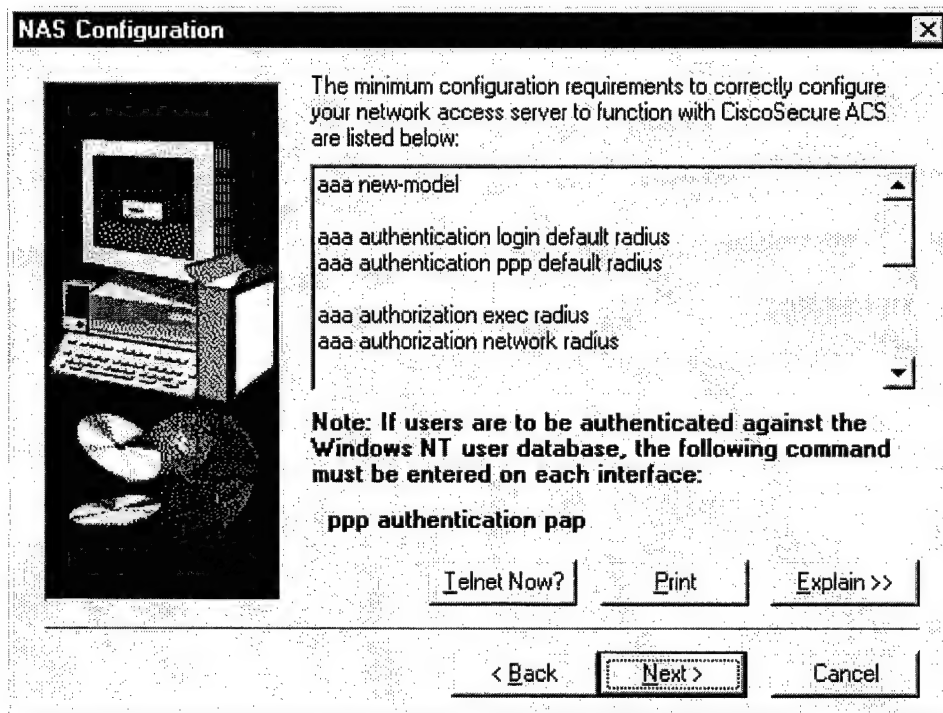
### RADIUS Installation and Maintenance Guide

---

An Access Server Configuration screen will appear (not pictured). This will prepare to establish a telnet session with the NAS and change the configuration. It will describe what is going to happen next. Click Next.

A NAS Configuration Screen is depicted in **Figure 17**. It will contain CiscoSecure commands to configure the NAS to communicate with the AAA Server.

1. Click "Telnet Now?". When this button is clicked, the information within this window is posted to the Windows Clipboard. It will also open a HyperTerminal window. Paste the configuration within the router HyperTerminal screen and type "exit".
2. Return to the NAS Configuration screen and click Next.



**Figure 17 - CS AAA Configuration Screen**

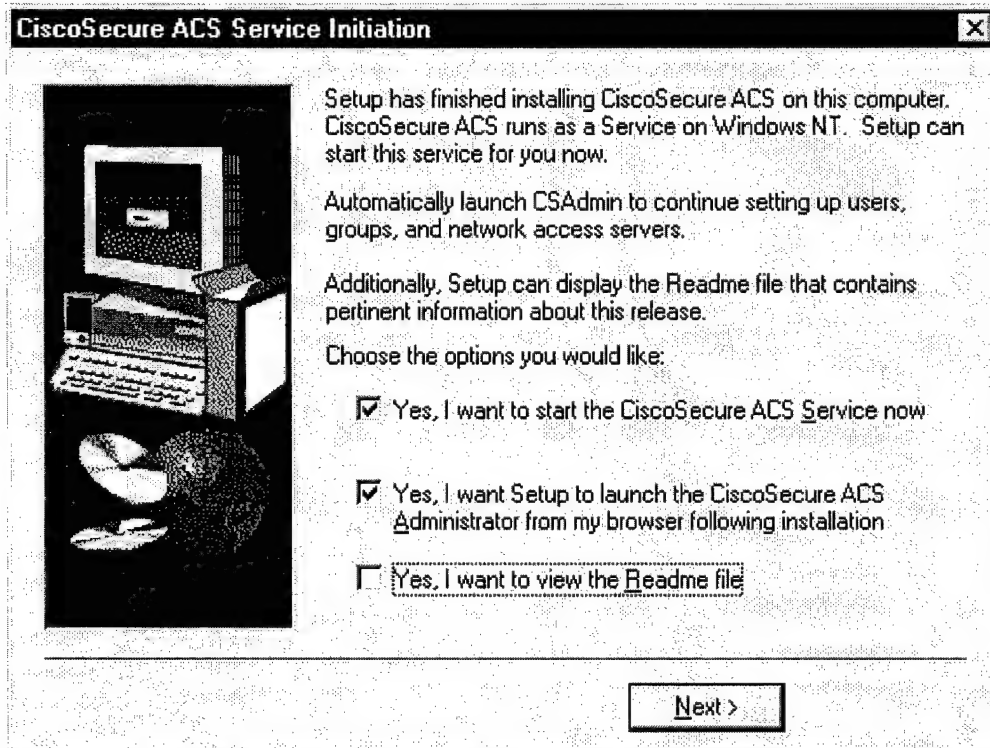
## DHIAP

### RADIUS Installation and Maintenance Guide

---

A CiscoSecure ACS Service Initiation screen is displayed in **Figure 18**.

1. Place a check in the check boxes for:
  - “Yes, I want to start the CiscoSecure ACS Service now”
  - “Yes, I want Setup to launch the CiscoSecure ACS Administrator from my browser following installation”
2. Click Next and the installation process will finish.



**Figure 18 - CS Service Initiation Screen**

#### **4.2 Configuring CiscoSecure Software**

After the CiscoSecure software has been installed, it will be necessary to configure it to authenticate, authorize and account dial-in users. There should be an administrative icon for the CiscoSecure Software on the Server Desktop. Click this icon to start the administrative configuration.

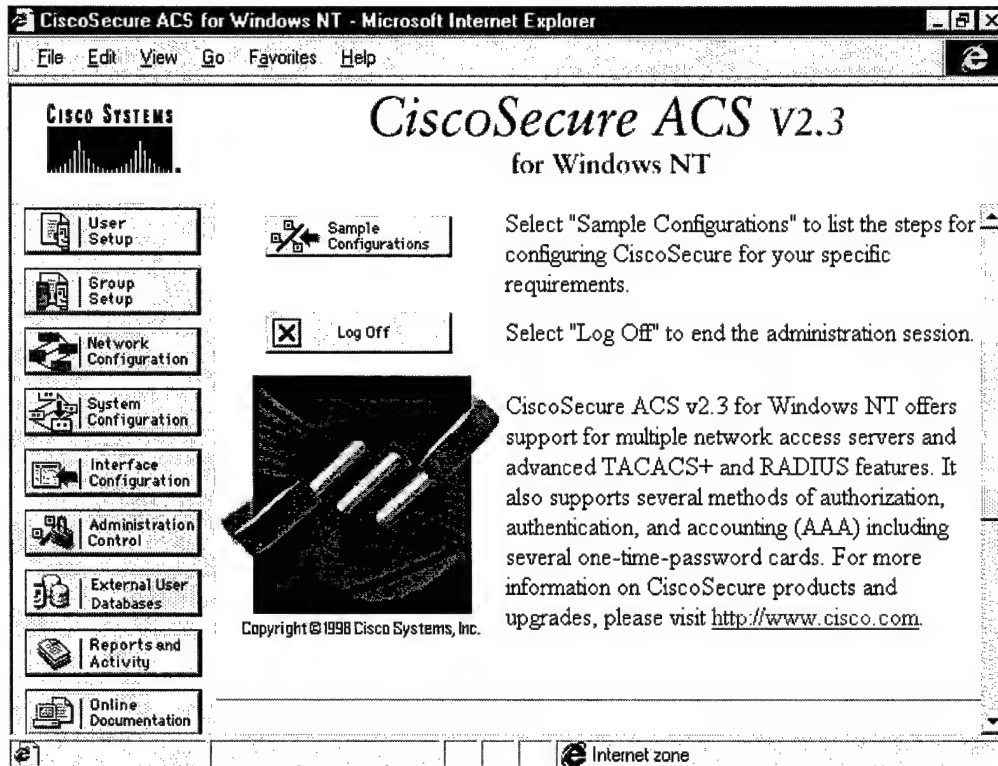


## DHIAP

### RADIUS Installation and Maintenance Guide

---

**Figure 19** displays the initial GUI configuration screen for CiscoSecure. There are nine configuration buttons on the left side of the browser. If all nine buttons cannot be seen, readjust the resolution of the monitor or remove indication bars (e.g., address and button) from the browser's interface.



**Figure 19 - CS GUI Interface Screen**

#### **Troubleshooting Tip: CiscoSecure Interface**

##### **Problem:**

Access to the CiscoSecure administrative software is gained through a browser. An icon is positioned on the Windows NT desktop to activate this process. Once activated, the browser performs as if it is navigating the World Wide Web. A technology called caching is used to quickly access an HTML page. This process stores in memory the last page accessed of a specific site. When that page is requested again, to save time, the cached copy will be displayed and not the most current page. This problem has been noted when adding, deleting, or making specific changes to users and groups in the CiscoSecure system. When the area of change is requested immediately after the change, it appears that no changes have been made at all. This is due to the cached copy of the initial page being used.

##### **Solution:**

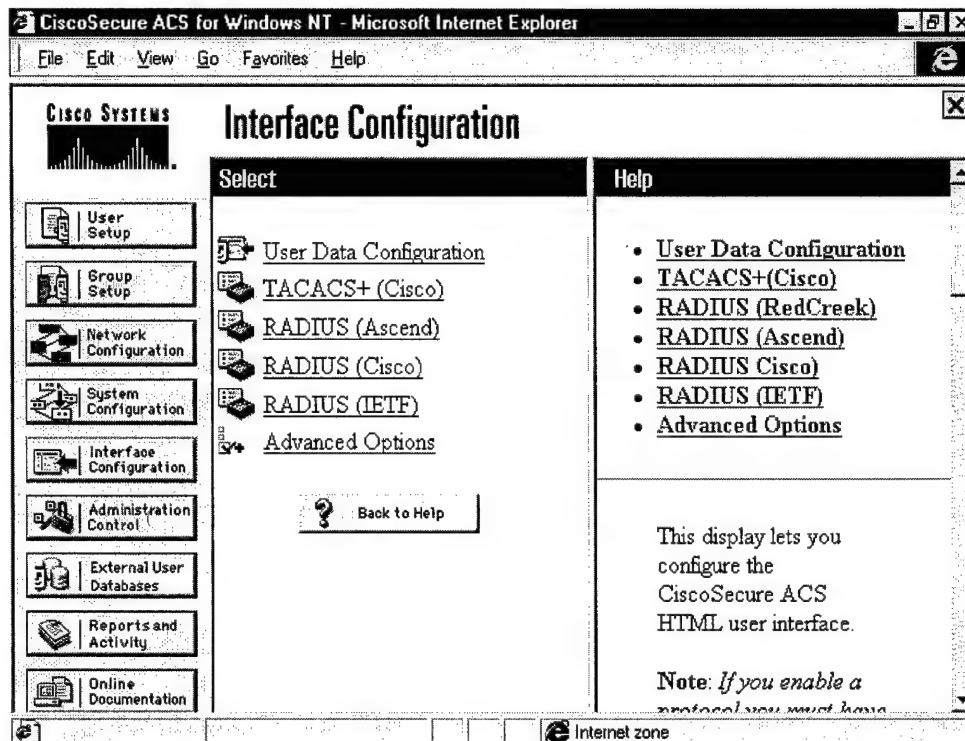
There are no noted system changes for this problem. However, click on the requested item again and the changes will be reflected.

# DHIAP

## RADIUS Installation and Maintenance Guide

### 4.2.1 Interface Configuration

Configure the Interface Configuration button first. This will establish the appropriate interface and system options in all system categories. The Interface Configuration button will allow the system administrator to provide the configuration options to be selected for all other categories. When clicked, the administrator should see the options listed in **Figure 20**.



**Figure 20 - CS Interface Configuration Screen**

Follow each hyperlink in the select form to the appropriate window and select the following information:

1. User Data Configuration  
Accept all defaults.
2. RADIUS (IETF)  
Accept all defaults.
3. RADIUS (Cisco)  
Click in the check box for "Group" [026] Vendor-Specific.  
The "User" can be left blank.  
Click "Submit."
4. Advanced Options  
Click all of the check boxes.  
Click "Submit."

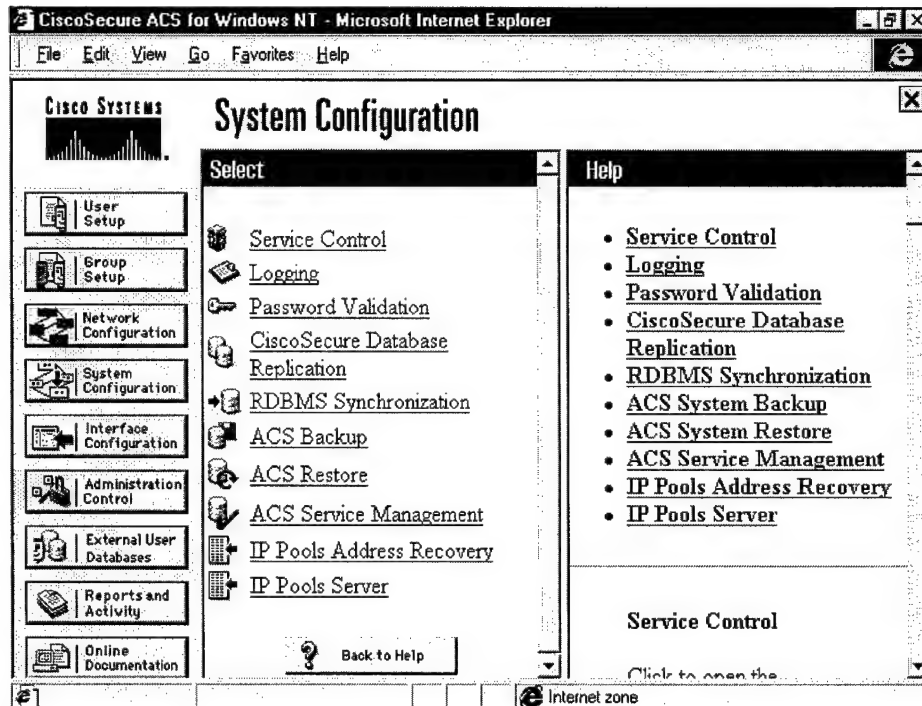
# DHIAP

## RADIUS Installation and Maintenance Guide

### 4.2.2 System Configuration

Select the System Configuration menu next. It will allow the administrator to change reporting attributes and system settings.

**Figure 21** displays the initial System Configuration access screen. To configure this section, edit the areas listed in the Select screen.



**Figure 21 - CS System Configuration Screen**

#### 4.2.2.1 Service Control

Service Control is an option for restarting the CiscoSecure Software when any changes have been made to the system. The following options can also be configured:

##### Service Log File Configuration

1. Select the level of detail that is appropriate for the location's needs.
2. Click Restart.

#### 4.2.2.2 Logging

Logging reports on the information configured in each of its settings. When Logging is clicked, no less than the following should appear:

##### Failed Attempts

1. Select Enable logging.
2. Select Columns to log.
3. Select attributes appropriate for location.
4. Log File Management.
5. Select attributes appropriate for the location.
6. Click Submit when finished.

## DHIAP

### RADIUS Installation and Maintenance Guide

---

#### RADIUS Accounting

1. Select Enable logging.
2. Select Columns to log.
3. Select attributes appropriate for the location.
4. Log File Management.
5. Select Attributes appropriate for the location.
6. Click "Submit" when finished.

If other categories are present:

1. Click the hyperlink to that area.
2. Click the box to "Disable Logging."
3. Click the "Submit" button.

#### **4.2.2.3 Password Validation**

Password Validation will enable the administrator to set password attributes. The following should be selected:

1. Password length can be 4-32 characters. [Adhere to local password policy.]
2. Password may not contain the username.
3. Password is different from the previous value.
4. Password must be alphanumeric.
5. Click "Submit" when finished.

#### **4.2.2.4 ACS Service Management**

Accept defaults. This option was configured during the initial setup.

#### **4.2.2.5 IP Pools Address Recovery**

IP Pools Address Recovery allows the administrator to set a time limit on allocated IP Addresses. If a user has been assigned an IP Address for longer than the designated period of time, the IP will be rescinded. Edit the appropriate length of time for the location's needs. Click "Submit" when finished.

#### **4.2.2.6 IP Pools Server**

IP Pools are named pools of IP Addresses that are dynamically allocated to dial-in users. When the dial-in user disconnects, the address is then placed back in the pool for reassignment. The system administrator must set this pool. **Figures 22 and 23** display the pool setup screens. To create a pool, click the "Add/Entry" button and enter the following:

1. Pool Name
2. Start/End Address
3. Click "Submit" when finished.

A pool can be edited by clicking on a pool name hyper-link.

# DHIAP

## RADIUS Installation and Maintenance Guide

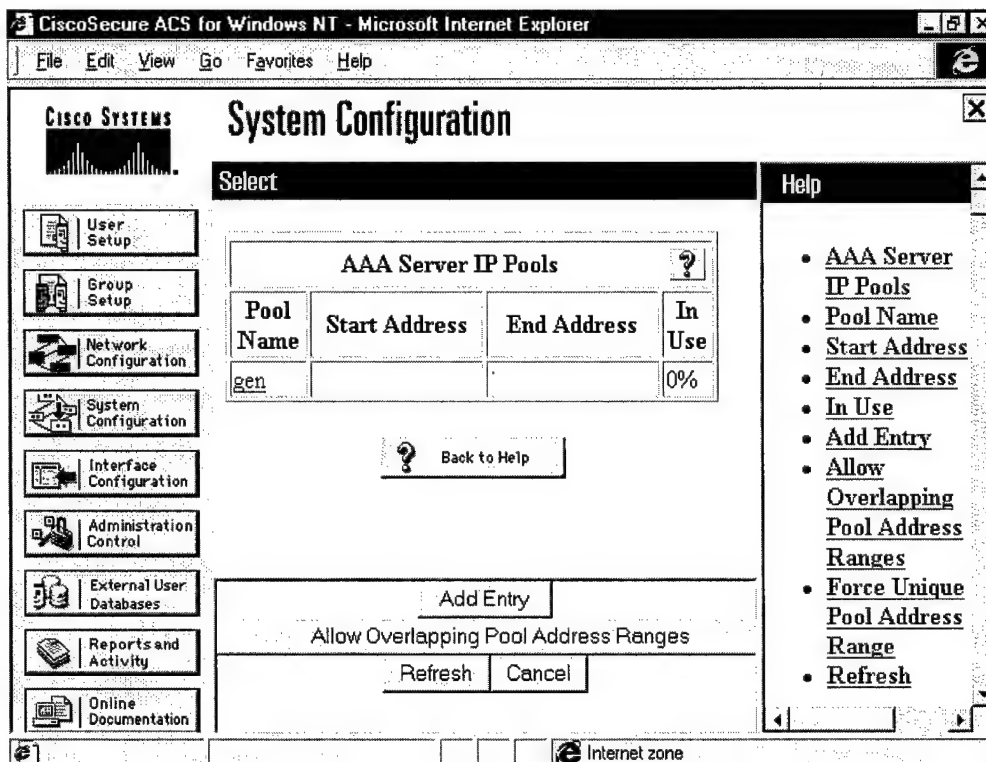


Figure 22 - CS System Configuration (AAA Server IP Pools) Screen

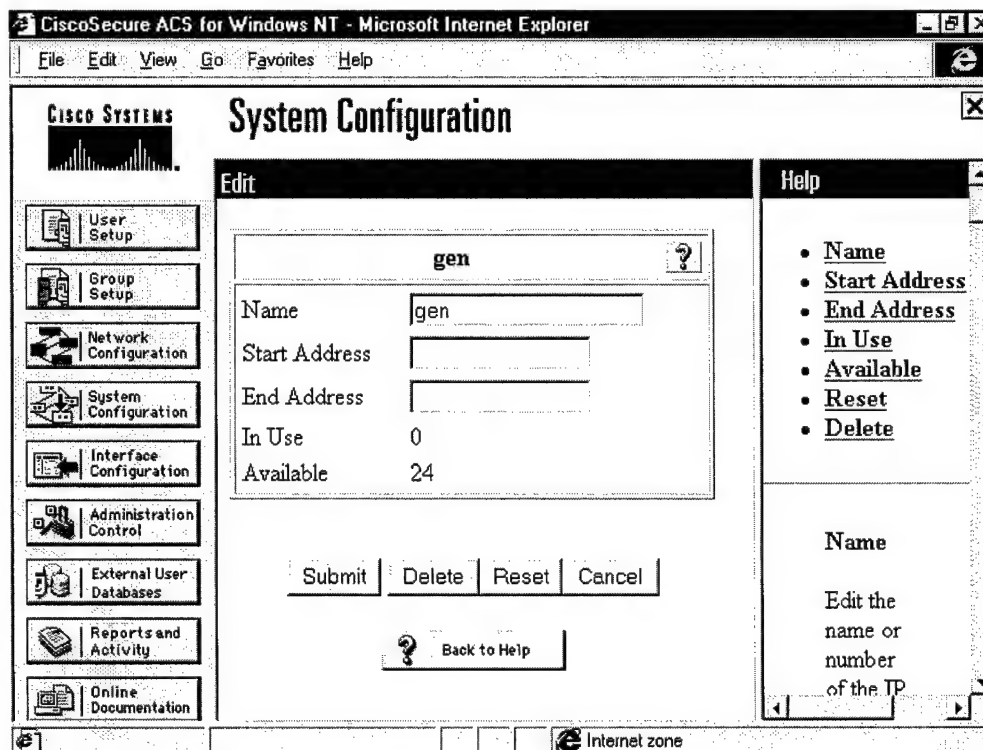


Figure 23 - CS System Configuration (Submit) Screen

## DHIAP

### RADIUS Installation and Maintenance Guide

---

#### 4.2.3 Group Setup

Group Setup will allow the administrator to combine users into several convenient groups. Later, users can be assigned to groups based on access permissions. Once entered, the attributes of the group can be changed, affecting all members of that group. There are three areas to focus on: Time of Day Login; IP Address Assignment; and RADIUS Attributes. To edit or configure a group, click the Group Setup button. Click the down arrow and select the group to edit or configure as displayed in **Figure 24**. Additional areas of configuration are displayed in **Figures 25, 26** and **27**. For initial configuration, click Edit Settings and configure the following:

1. Voice-Over-IP Support  
N/A
2. Default Time of Day Access Settings (see **Figure 25**)  
Click the check box and drag the mouse over the appropriate times. This setting will allow the administrator to set the login hours permitted for a remote dial-in user.
3. Callback  
The callback feature allows reversal of phone charges by calling the dial-in user back. The Dialup Client specifies callback number. If Windows NT authentication will be used, check to ensure that this feature is selected in the User Manager, account name, dial-in access configuration boxes.
4. Network Access Restrictions  
None are needed at this time for NAS or Telnet.
5. Max Sessions  
As appropriate for location standards.
6. Password Aging Rules  
This is applicable to CiscoSecure database members only. Enter as appropriate for the location standards.
7. Generate Greetings (for successful logins)  
None needed.
8. IP Assignment (see **Figure 26**)  
Click the radio button "Assigned from AAA server pool".  
Place the appropriate pool in the Selected Pools box.  
NOTE: This option must be preconfigured in the System Configuration – IP Pools Server section of this utility.

## DHIAP

### RADIUS Installation and Maintenance Guide

---

9. IETF RADIUS Attributes (see **Figure 27**) - Select:

[006] Framed

[007] PPP

[011] Type – (**150.in**) [This will allow access-list 150 to be applied to all incoming calls. Access Lists were discussed in Section 3. They are crucial for the implementation of restrictions.]

All other options in this category should be left as default.

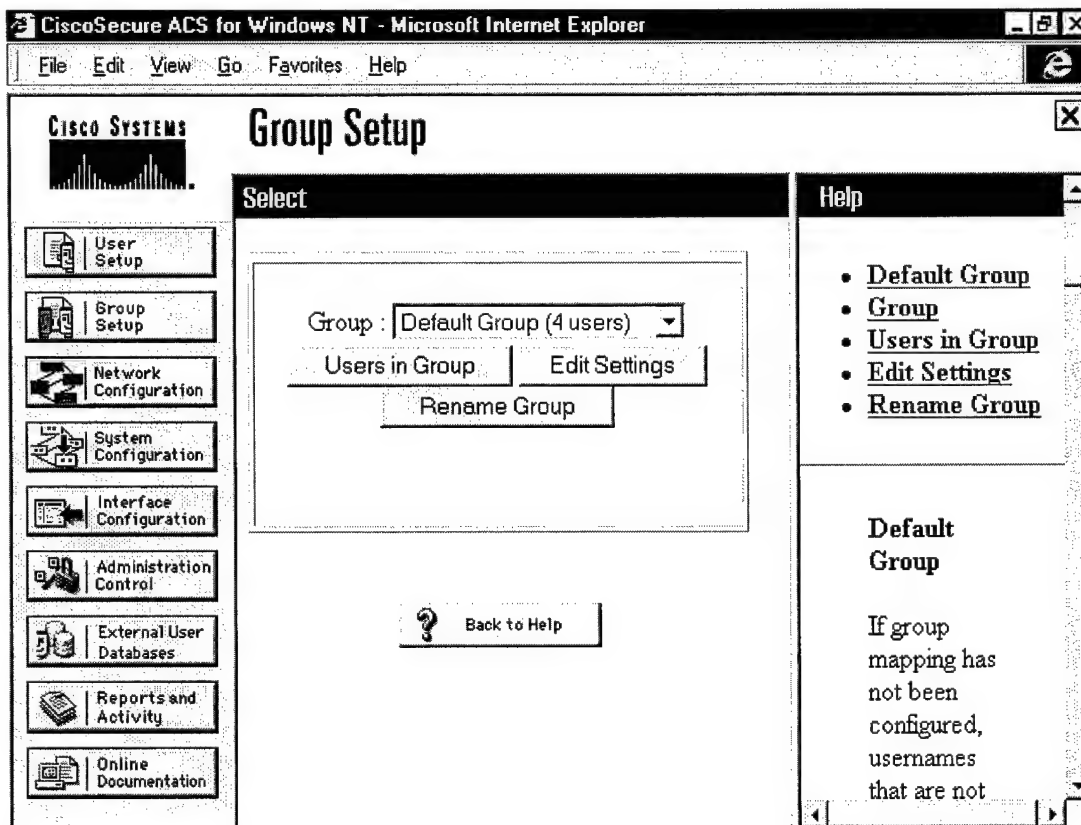
10. Cisco RADIUS Attributes - Select:

[009\001] AV –Pair

Note: This filter needs to be applied only if a specific callback number will be called every time a user calls in. The text should read:

Lcp:callback-dialstring=(insert callback number specific to user)

When all settings are applied, click “Submit and Restart.”



**Figure 24 - CS Group Setup Screen**

# DHIAP

## RADIUS Installation and Maintenance Guide

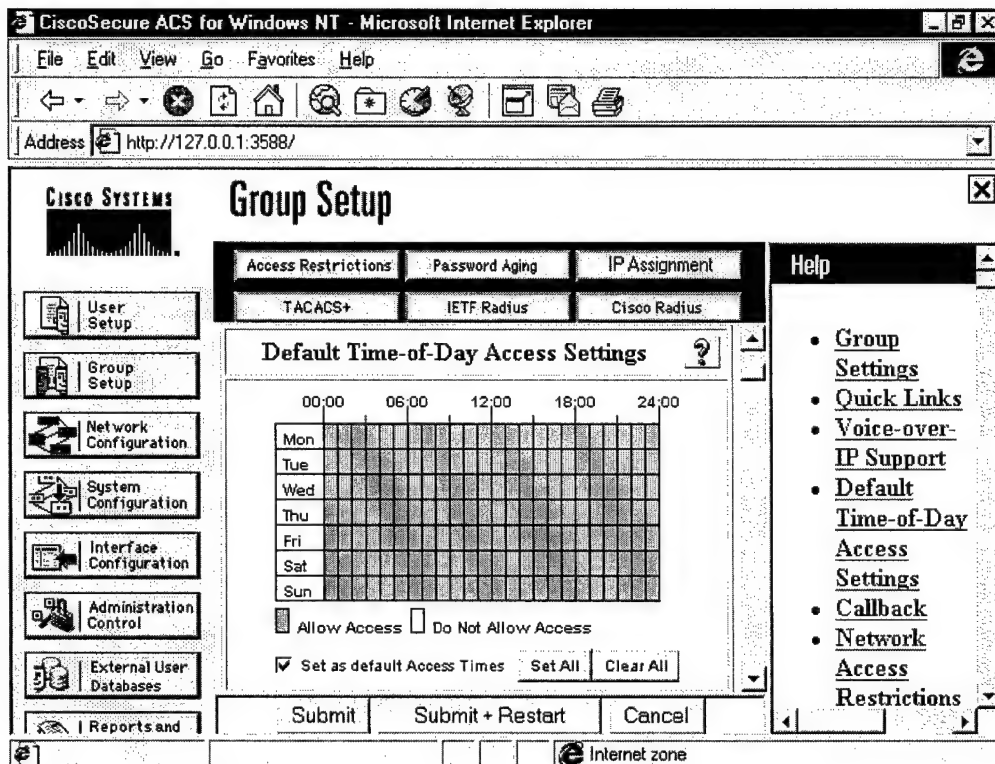


Figure 25 - CS Group Setup (Time-of-Day) Screen

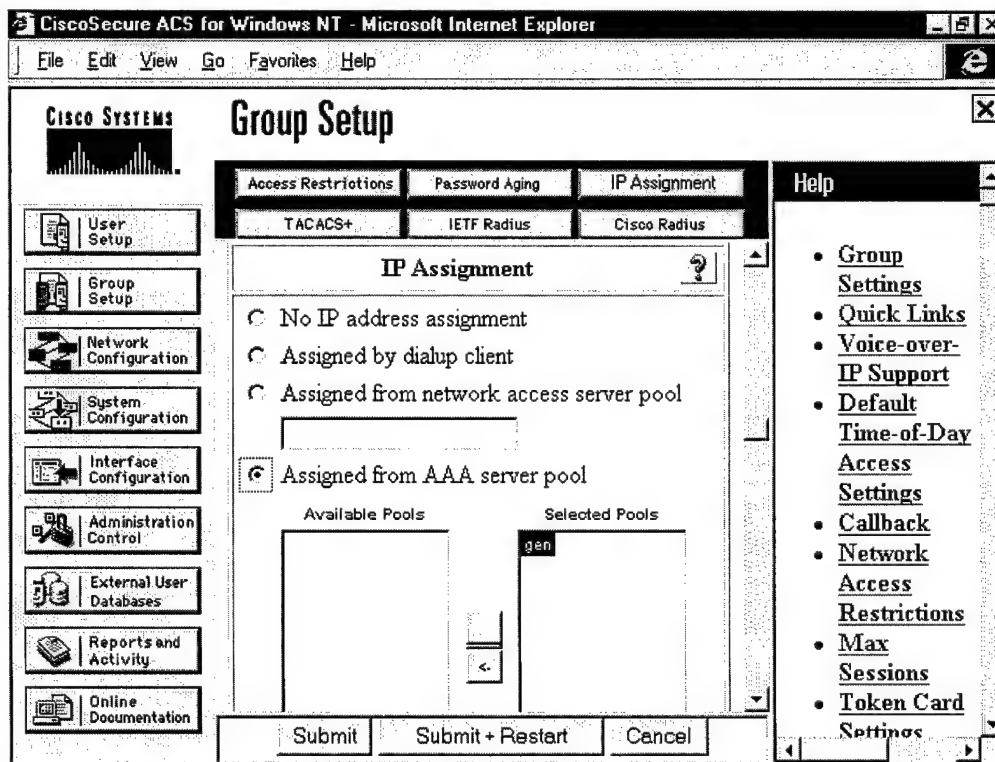


Figure 26 - CS Group Setup (IP Assignment) Screen



# DHIAP

## RADIUS Installation and Maintenance Guide

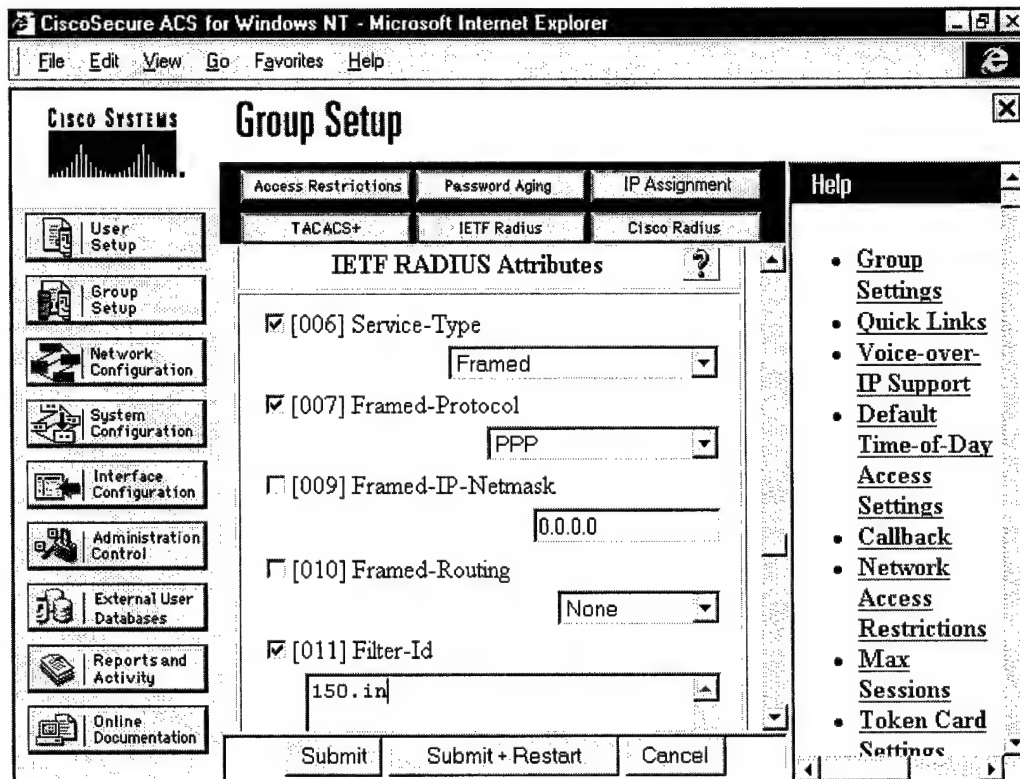


Figure 27 - CS Group Setup (IETF RADIUS Attributes) Screen

Options Not Pictured:

[027] Session-Time

[028] Idle-Time

The options listed above should be set in **seconds**. The time values should be determined by each site during installation.

Note Troubleshooting Tip, Client Disconnect in Section 5.3.2.

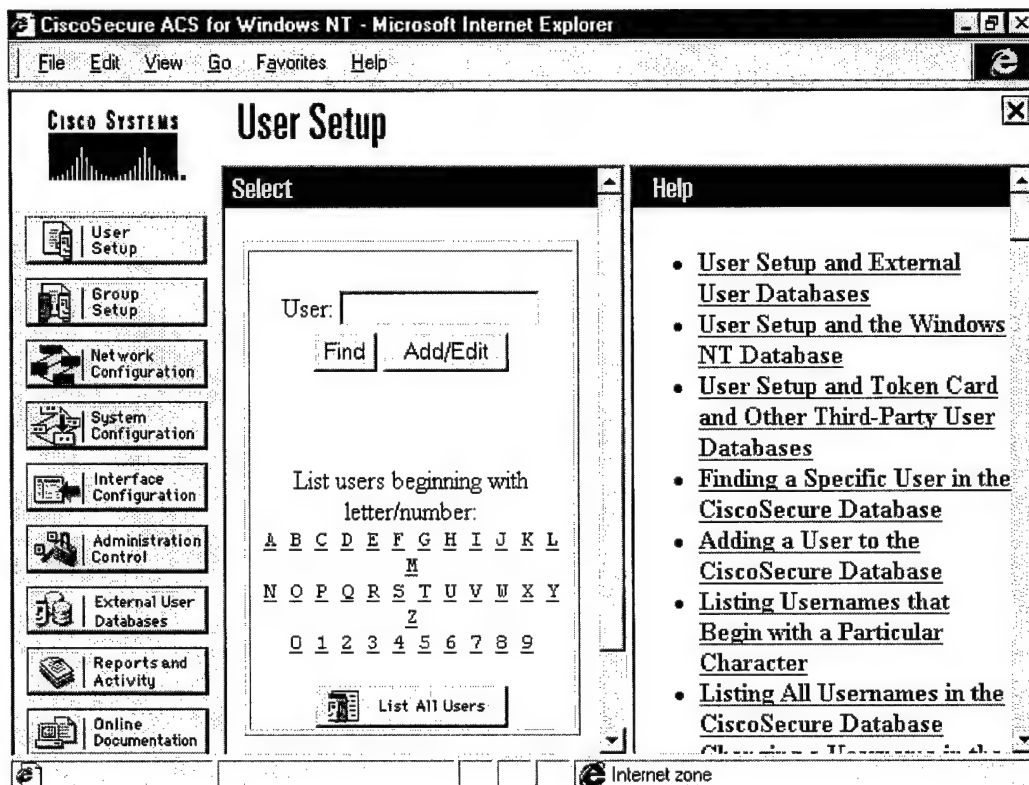
## DHIAP

### RADIUS Installation and Maintenance Guide

---

#### 4.2.4 User Setup

The User Setup button will allow the administrator to create/delete users and modify their profiles. Within the RADIUS compliant system there are two types of dial-in users. The first is a CiscoSecure database user and the second is a Windows NT database user. CiscoSecure database users are created by an administrator within the CiscoSecure administrative software and assigned to an appropriate group with access restrictions imposed on that group. Windows NT users are not originally members of the CiscoSecure database. They become members after successfully dialing into the system and being authenticated. CiscoSecure automatically takes the authenticated Windows NT users and place them into a default group within the CiscoSecure database. The administrator can then move the users from the default group into an appropriate group with access restrictions imposed on that group. Both types of users can be administered from the CiscoSecure administrative software. **Figure 28** displays the initial User Setup screen.



**Figure 28 - CS User Setup Screen**

## DHIAP

### RADIUS Installation and Maintenance Guide

---

User Setup for Password and Group Assignment are displayed in **Figures 29** and **30**. To add a user, type the user's name or remote ID in the space provided and click Add/Edit. Once created, enter the following information:

1. Real Name
2. User Setup
  - CiscoSecure Password or leave blank if the name and password will be derived from a Win NT database.
  - For Password Authentication, click Windows NT or CiscoSecure, whichever applies.
3. Group Mappings
  - Assign the user to a group or a external database.
4. Callback
  - Select "Use group setting".
5. Client IP Address Assignment
  - Select "Use group setting".
6. Advanced Settings
  - None
7. Max Sessions
  - Select "Use group setting".
8. Account Disable
  - Fill in as appropriate to location standards.
9. Click Submit to finish.

# DHIAP

## RADIUS Installation and Maintenance Guide

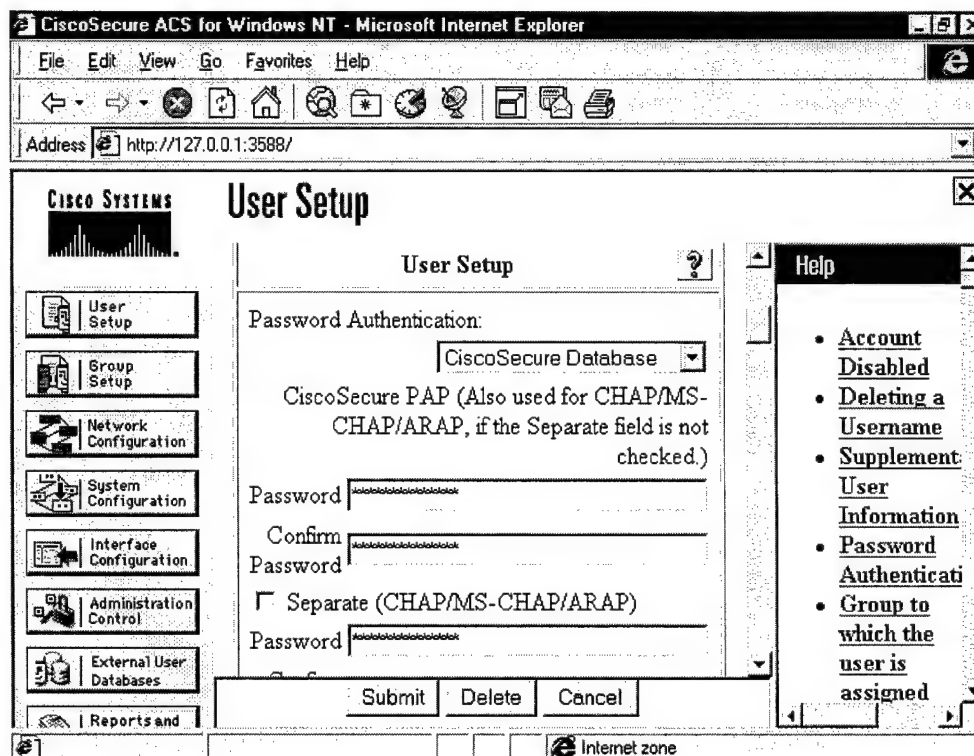


Figure 29 - CS User Setup (Passwords) Screen

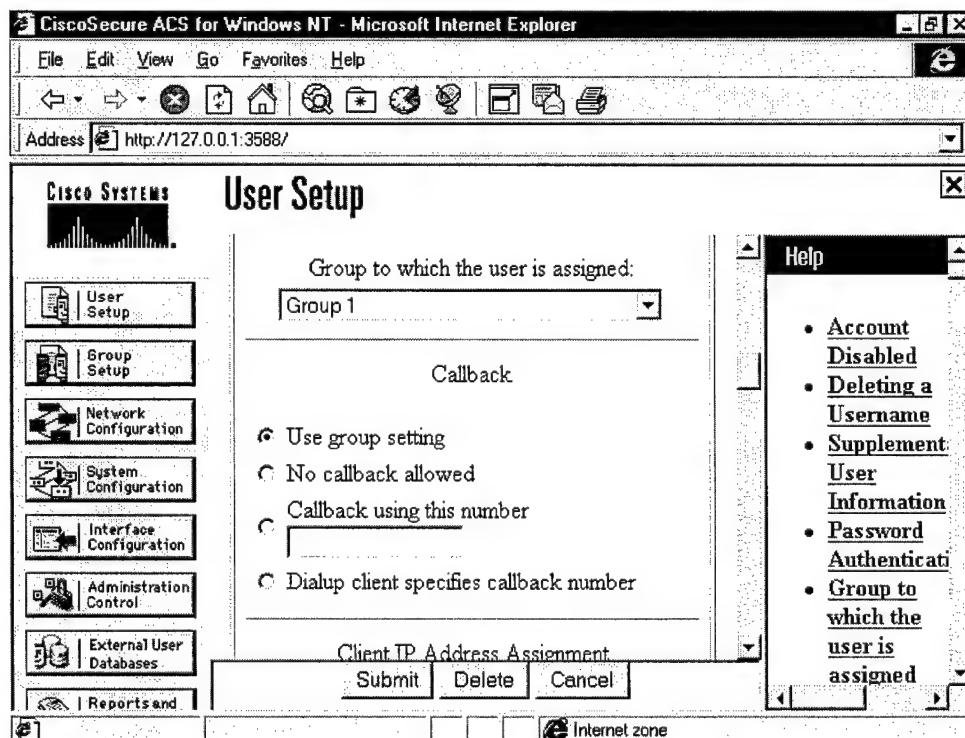


Figure 30 - CS User Setup (Groups) Screen

## DHIAP

### RADIUS Installation and Maintenance Guide

---

#### **Troubleshooting Tip: Windows NT Duplicate User**

##### **Problem:**

Dialing into the CiscoSecure RADIUS compliant system can be accomplished from clients using Windows 95/98 and Windows NT 4.0 operating systems. When the dial-in client is authenticated, an entry is made in the CiscoSecure logs pertaining to that session. The first time a Windows NT user logs into the system, their User ID is placed into a default group of the CiscoSecure database. The entry is made in the database consisting of the "domain name\user id". For instance, if the user was User1 of the DHIAP domain, the database entry would appear as "DHIAP\User1". If that same user dialed in via a Windows 95/98 client, the database entry would consist of only the "user id". For instance, if the user was User1, the database entry would appear as "User1". The end result would be two separate and different entries into the database for one user. This problem is due to Windows NT having the capability to automatically provide the CiscoSecure system with a domain name. The Windows 95/98 dial-in client does not have this capability.

##### **Solution:**

To correct this problem, the system administrator must create an automatic domain prefix in the Network Configuration, Distribution Area of the CiscoSecure graphical user interface. For a DHIAP domain, the "character string" entry should appear as "DHIAP\". Its prefix should be set to "yes" and the appropriate authenticating server attached to the prefix. **Figures 31** and **32** are provided as an example. Once the problem has been corrected, delete the user created with the original domain name and user id, i.e. "DHIAP\User1".

##### **Analysis:**

By adding the domain prefix to a specified authenticating server, the problem of duplicate User ID entries will be eliminated. These prefixes can be applied to multiple domains and multiple servers configured in the system.

# DHIAP

## RADIUS Installation and Maintenance Guide

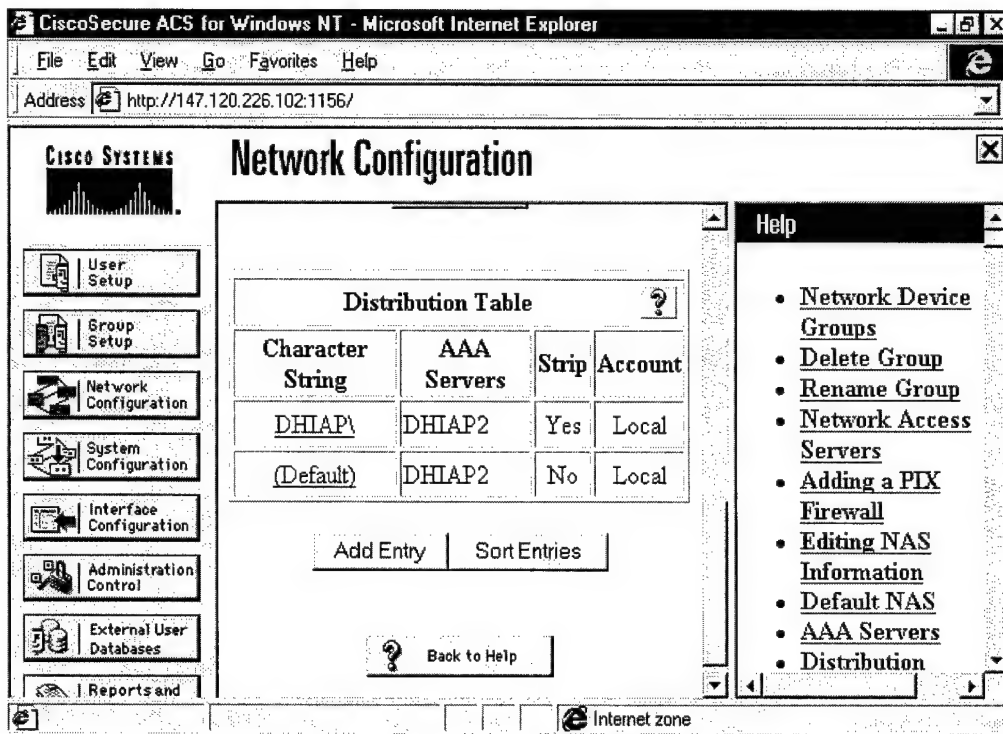


Figure 31 - Distribution Table Screen

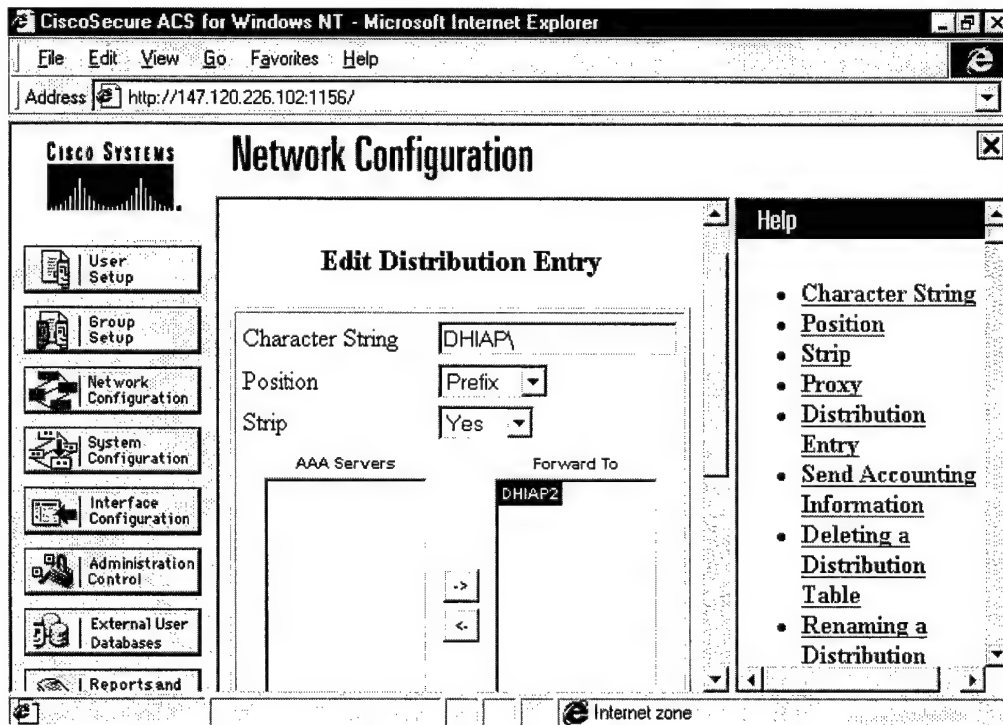


Figure 32 - Distribution Area Editing Screen

# DHIAP

## RADIUS Installation and Maintenance Guide

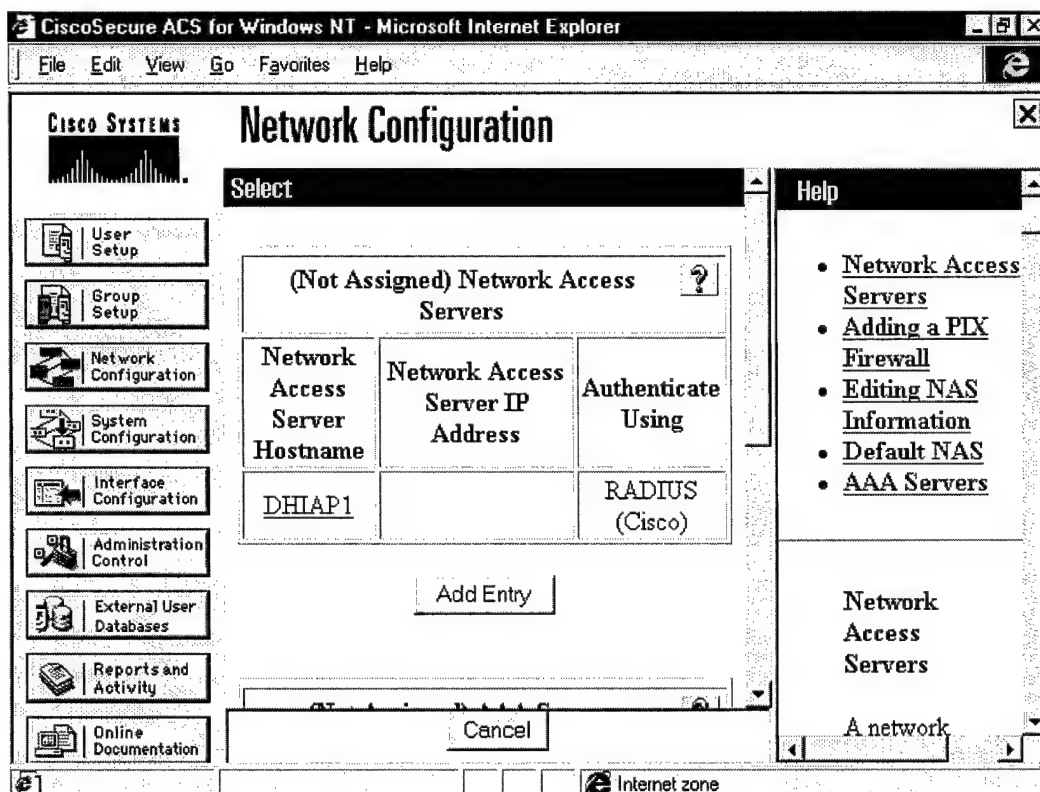
### 4.2.5 Network Configuration

Network Configuration will allow the administrator to set up Network Device Groups which are groups of Network Access Servers, and AAA Servers. When the Network Configuration button is clicked, the administrator will either see a Network Device Group box or a Network Access Server box. If the Network Device Group box appears, click on the Network Device Group link. From this point, Network Access Servers and AAA Servers can be configured and entered.

**Figure 33** displays the initial Network Configuration screen.

To configure the Network Configuration area, go to the Network Access Servers box (see **Figure 34**):

1. Click the NAS Hostname link to reconfigure OR Click Add Entry for initial configuration.
2. Enter the NAS Name and IP Address (for the router). [For illustration purposes, DHIAP1 is being used for the NAS.]
3. Enter the secret key for communications between the NAS and CiscoSecure Server.
4. Authenticate Using RADIUS (Cisco).
5. When complete, click Submit and Restart.



**Figure 33 - CS Network Configuration Screen**

## DHIAP

### RADIUS Installation and Maintenance Guide

---

**Figure 34** is the configuration screen that appears when the Network Access Server link is clicked.

**CiscoSecure ACS for Windows NT - Microsoft Internet Explorer**

File Edit View Go Favorites Help

**CISCO SYSTEMS**

**Network Configuration**

**Access Server Setup For DHIAP1**

Network Access Server IP Address: 147.120.226.101

Key:

Network Device Group:

Authenticate Using: RADIUS (Cisco)

☐ Single Connect TACACS+ NAS (Record stop in accounting on failure).

☐ Log Update/Watchdog Packets from this Access Server

**Help**

- [Network Access Server IP Address](#)
- [Key](#)
- [Network Device Group](#)
- [Authenticate Using](#)
- [Single Connect TACACS+ NAS](#)
- [Log Update/Watchdog Packets from this Access Server](#)
- [Relating an](#)

**Figure 34 - CS Network Configuration (Server Setup) Screen**

**Figure 35** is the CiscoSecure Server Configuration screen.

**CiscoSecure ACS for Windows NT - Microsoft Internet Explorer**

File Edit View Go Favorites Help

**CISCO SYSTEMS**

**Network Configuration**

**(Not Assigned) AAA Servers**

AAA Server Name	AAA Server IP Address	AAA Server Type
<a href="#">DHIAP3</a>		CiscoSecure ACS for Windows NT
<a href="#">DHIAP2</a>		CiscoSecure ACS for Windows NT

[Add Entry](#)

[Back to Help](#)

[Cancel](#)

**Help**

- [Network Access Servers](#)
- [Adding a PIX Firewall](#)
- [Editing NAS Information](#)
- [Default NAS](#)
- [AAA Servers](#)

**Network Access Servers**

[A network](#)

**Figure 35 - CS Network Configuration (AAA Servers) Screen**



# DHIAP

## RADIUS Installation and Maintenance Guide

Figures 36 and 37 are the CiscoSecure Server Configuration screens for a specific server.

The screenshot shows the CiscoSecure ACS for Windows NT - Microsoft Internet Explorer window. The title bar reads "CiscoSecure ACS for Windows NT - Microsoft Internet Explorer". The menu bar includes "File", "Edit", "View", "Go", "Favorites", and "Help". The main content area is titled "Network Configuration" and "AAA Server Setup For DHAIP3". On the left, there is a vertical navigation pane with icons and labels: "User Setup", "Group Setup", "Network Configuration" (highlighted), "System Configuration", "Interface Configuration", "Administration Control", "External User Databases", "Reports and Activity", and "Online Documentation". The main configuration area contains the following fields and options:

- AAA Server IP Address: [Text Field]
- Key: [Text Field]
- Network Device: (Not Assigned) [Dropdown]
- Group: [Text Field]
- ☐ Log Update/Watchdog Packets from this remote AAA Server
- AAA Server: CiscoSecure ACS for Windows NT [Dropdown]

On the right, there is a "Help" pane with a list of links:

- [AAA Server IP Address](#)
- [Key](#)
- [Network Device Group](#)
- [Log Update/Watchdog Packets from this Remote AAA Server](#)
- [AAA Server Type](#)
- [Traffic Type](#)

The status bar at the bottom indicates "Internet zone".

Figure 36 - CS Network Configuration (AAA Server Setup) Screen

The screenshot shows the same CiscoSecure ACS for Windows NT - Microsoft Internet Explorer window, but with the "Network Configuration" screen updated. The title bar and menu bar are the same. The main content area is titled "Network Configuration" and "AAA Server Setup For DHAIP3". The left navigation pane is the same. The main configuration area now includes:

- Device: (Not Assigned) [Dropdown]
- Group: [Text Field]
- ☐ Log Update/Watchdog Packets from this remote AAA Server
- AAA Server: CiscoSecure ACS for Windows NT [Dropdown]
- AAA Type: [Text Field]
- Traffic Type: inbound/outbound [Dropdown]

Below the configuration fields, there are four buttons: "Submit", "Submit + Restart", "Delete", and "Cancel". At the bottom, there is a "Back to Help" button with a question mark icon. The status bar at the bottom indicates "Internet zone".

Figure 37 - CS Network Configuration (Submit and Restart) Screen

# DHIAP

## RADIUS Installation and Maintenance Guide

---

### 5 REMOTE CLIENT CONFIGURATION

#### 5.1 Hardware Requirements

Workstations will need access to a phone line and a modem.

Laptops will need a modem, cable and phone line.

#### 5.2 Software Requirements

CiscoSecure will support Windows 95/98 or Windows NT 4.0 client software.

#### 5.3 Workstation Configuration

Windows 95/98 and Windows NT Workstation can be used to dial into the CiscoSecure RADIUS compliant system. There are minor configuration differences between Win 9X and Win NT.

##### 5.3.1 Windows 95/98

TCP/IP will need to be configured for dial-up networking on Win 9X dial-in clients. Note: This task can be accomplished with the property settings on Network Neighborhood or in the Dial-Up Networking properties of the icon that has been set up to perform the remote dial. It is recommended that these properties be set up within the icon properties, as described below.

From the Icon Properties menu, go to:

1. General Tab

Insert the phone number of the Cisco Router to be dialed and configure the modem as necessary. It may be necessary to configure multiple phonebook entries in Dial-Up Networking for numbers such as local, long distance, etc.

2. Server Type Tab

For the type of server, select PPP, Windows NT Server, and Windows 95 or 98.

Advanced Options

Choose Log onto the network.

Allow Network Protocols

Choose TCP/IP.

TCP/IP Settings

Choose Server Assigned IP Address.

Select and assign the appropriate WINS Address.

Select and use IP Header Compression.

Select Use Default Gateway on Remote Network.

3. Scripting Tab

Allow defaults.

4. Multilink Tab

Allow defaults.

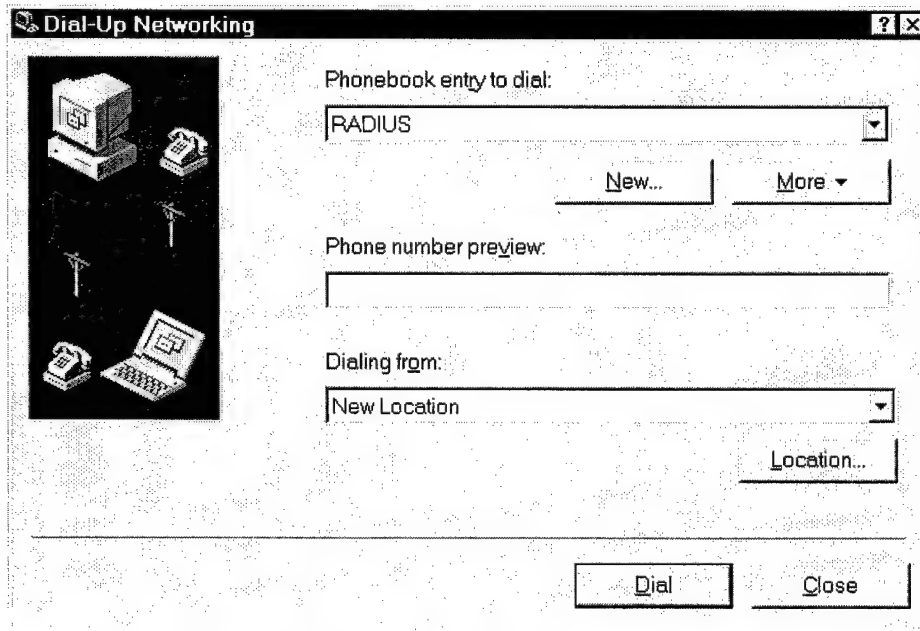
# DHIAP

## RADIUS Installation and Maintenance Guide

---

### 5.3.2 Windows NT 4.0

For Windows NT to effectively communicate with the CiscoSecure Software, standard Dial-Up Networking (DUN) must be configured on the workstation. This includes installing a modem and configuring it for standard use. Once Dial-Up Networking has been established, all configuration parameters for the workstation should be configured at the DUN configuration box. To start, click on the DUN menu item located in Programs and Accessories menus of the Start button. **Figure 38** displays the initial start screen.



**Figure 38 - Initial Dial-Up Networking Screen**

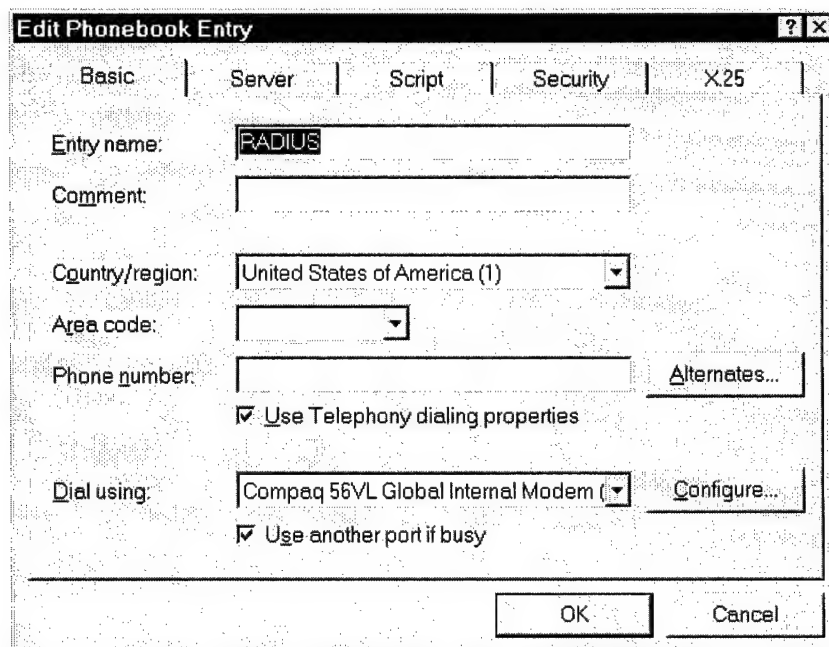
When the More button is clicked, a drop down menu will appear. Select the Edit Entry and Modem Properties section of the menu.

## DHIAP

### RADIUS Installation and Maintenance Guide

---

**Figure 39** displays the Edit Phonebook Entry screen.

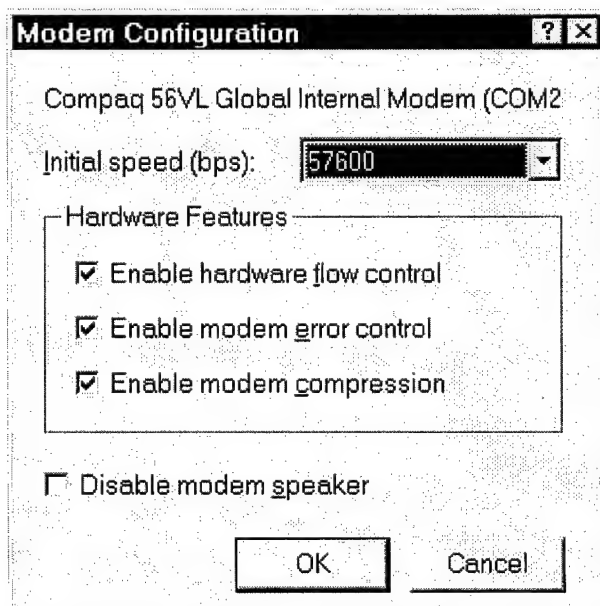


The screenshot shows the 'Edit Phonebook Entry' dialog box with the following fields and options:

- Tabbed interface: Basic (selected), Server, Script, Security, X.25
- Entry name:
- Comment:
- Country/region:
- Area code:
- Phone number:  Alternates...
- ☒ Use Telephony dialing properties
- Dial using:  Configure...
- ☒ Use another port if busy
- Buttons: OK, Cancel

**Figure 39 - Edit Phonebook Entry Screen**

In this screen, the call name should be entered as well as the number that is to be dialed. Modem properties under the Configure button should also be checked. **Figure 40** provides an example of the information needed for proper modem configuration.



The screenshot shows the 'Modem Configuration' dialog box with the following fields and options:

- Compaq 56VL Global Internal Modem (COM2)
- Initial speed (bps):
- Hardware Features:
  - ☒ Enable hardware flow control
  - ☒ Enable modem error control
  - ☒ Enable modem compression
- ☐ Disable modem speaker
- Buttons: OK, Cancel

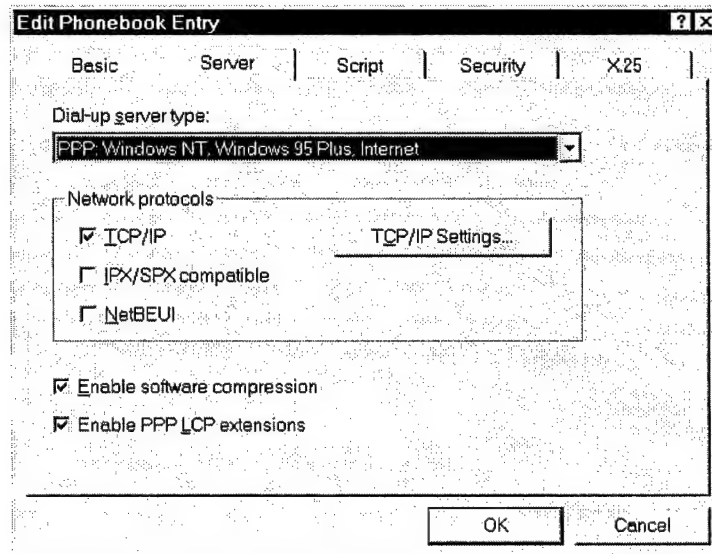
**Figure 40 - Modem Configuration Screen**

## DHIAP

### RADIUS Installation and Maintenance Guide

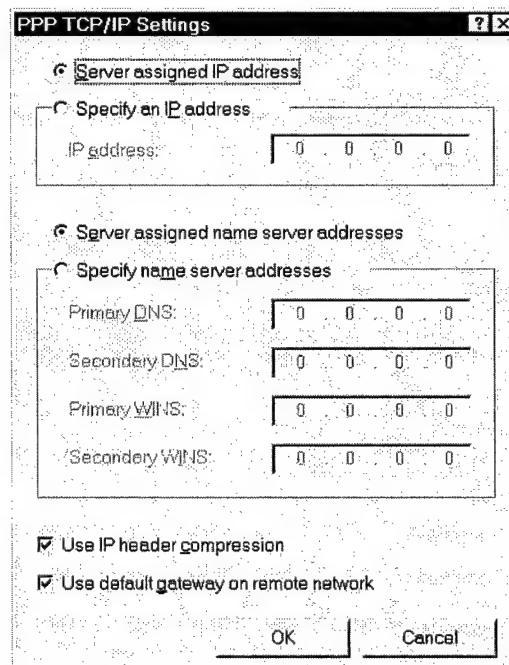
---

**Figure 41** displays the Server tab and its appropriate settings. As shown, select TCP/IP, Enable software compression and Enable PPP extensions.



**Figure 41 - Edit Phonebook Entry Screen – Server Tab**

Click on the TCP/IP Settings button. The screen in **Figure 42** is displayed. Make selections as shown.



**Figure 42 - PPP TCP/IP Settings Screen**

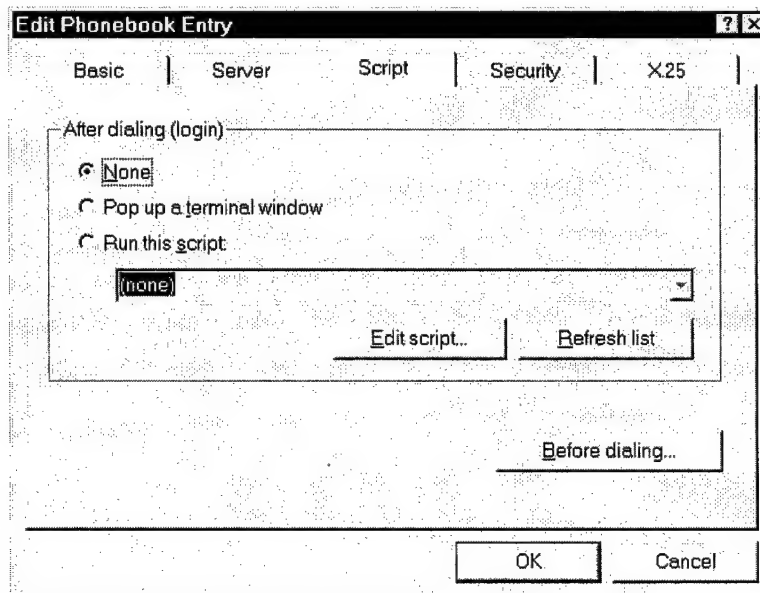
The Server should assign the IP Addresses and Name Server. Use IP header compression and default gateways on the remote network. Then click OK.

## DHIAP

### RADIUS Installation and Maintenance Guide

---

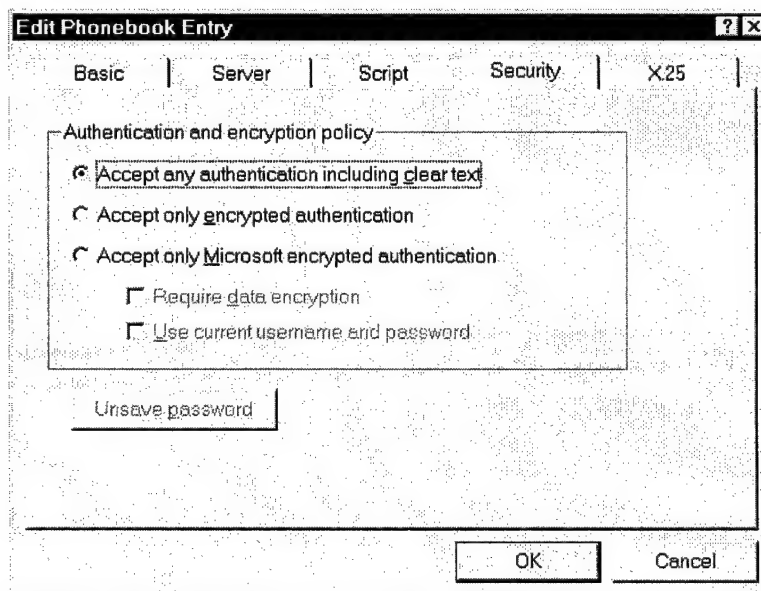
From the main screen, click on the Script tab as illustrated in **Figure 43**.



**Figure 43 - Edit Phonebook Entry Screen – Script Tab**

No scripts should be used at this time.

Click on the Security tab shown in **Figure 44**. Select the Authentication and encryption policy option, "Accept any authentication including clear text", that allows clear text to be used for authentication. **This selection is crucial.** If this portion is not correctly configured, the workstation will not be able to log into the RADIUS compliant system.



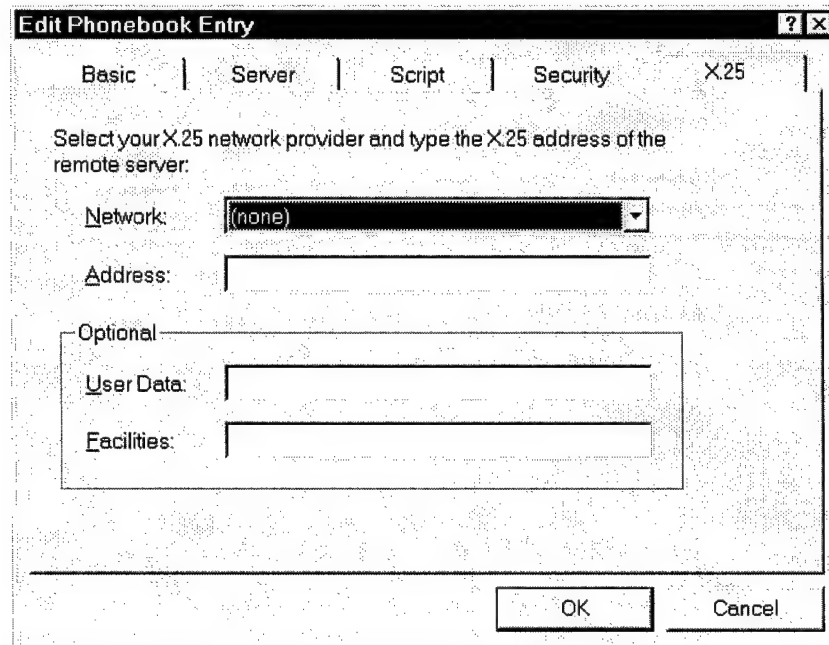
**Figure 44 - Edit Phonebook Entry Screen – Security Tab**

## DHIAP

### RADIUS Installation and Maintenance Guide

---

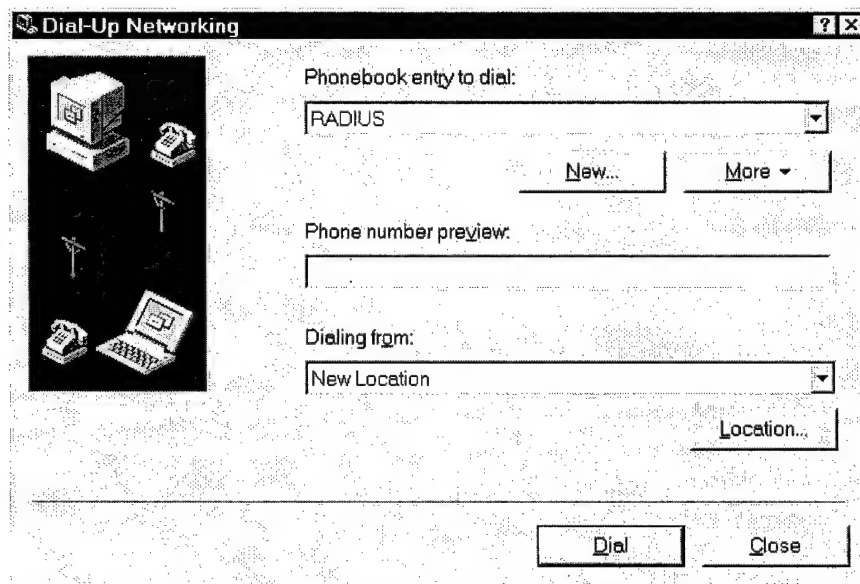
For completeness, **Figure 45** displays the default values for X.25 tab. The defaults are acceptable and will not adversely affect the operation of the workstation.



The image shows a Windows-style dialog box titled "Edit Phonebook Entry". It has five tabs: "Basic", "Server", "Script", "Security", and "X.25". The "X.25" tab is selected. Inside the dialog, there is a text label "Select your X.25 network provider and type the X.25 address of the remote server:". Below this, there is a "Network:" label followed by a dropdown menu showing "(none)". Below that is an "Address:" label followed by an empty text input field. A group box labeled "Optional" contains two more text input fields: "User Data:" and "Facilities:". At the bottom right of the dialog are "OK" and "Cancel" buttons.

**Figure 45 - Edit Phonebook Entry Screen X.25 Tab**

Once all tabs have been configured, click OK and return to the main menu as displayed again in **Figure 46**.



The image shows a Windows-style dialog box titled "Dial-Up Networking". On the left side, there is a vertical panel with four icons: a desktop computer with a modem, a telephone, a radio tower, and a laptop. To the right of this panel, there is a "Phonebook entry to dial:" label followed by a dropdown menu showing "RADIUS". Below this dropdown are "New..." and "More" buttons. Further down is a "Phone number preview:" label followed by an empty text input field. Below that is a "Dialing from:" label followed by a dropdown menu showing "New Location". To the right of this dropdown is a "Location..." button. At the bottom right of the dialog are "Dial" and "Close" buttons.

**Figure 46 - Dial-Up Networking Screen**

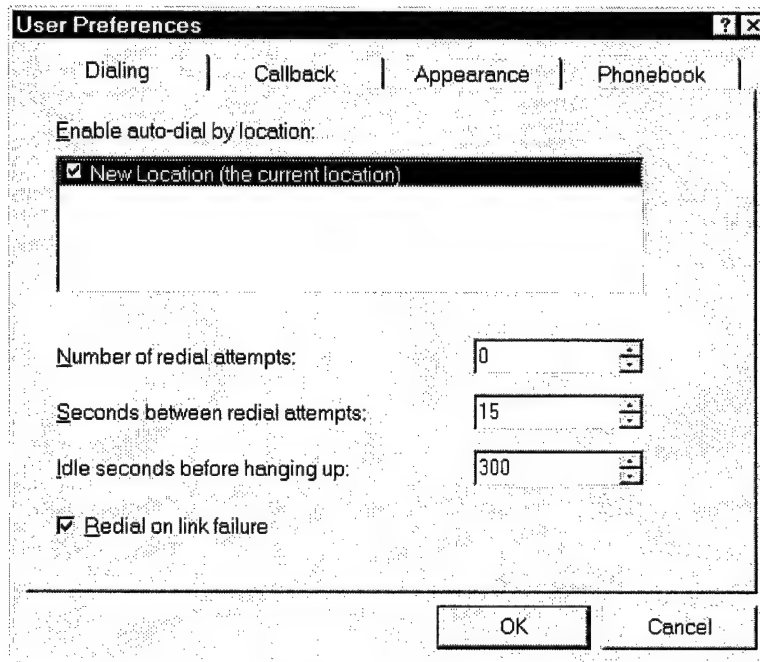
User preferences must also be set. From the More button, select User Logon Preferences.

## DHIAP

### RADIUS Installation and Maintenance Guide

---

**Figure 47** displays the appropriate settings for the Dialing tab; enter the required values as shown. All other tabs may contain the default information. Click OK.



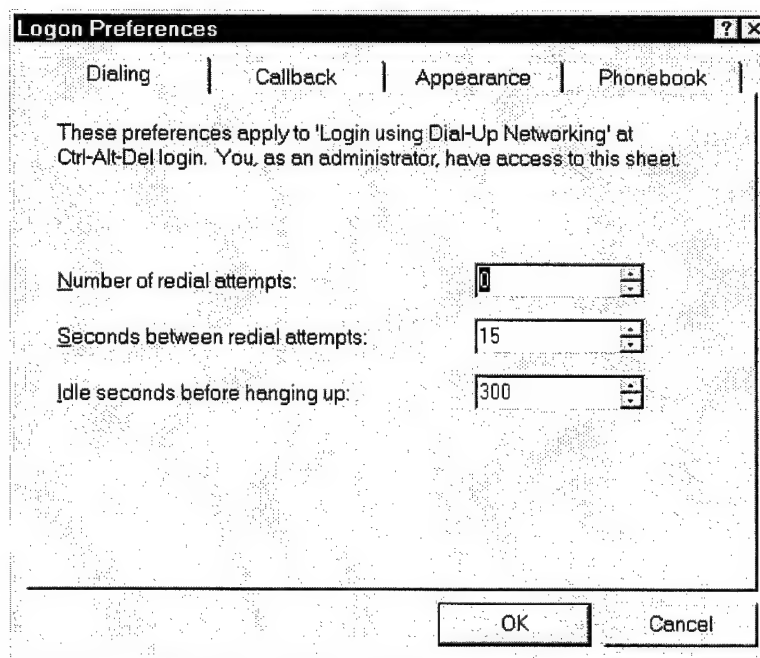
The 'User Preferences' dialog box has four tabs: 'Dialing', 'Callback', 'Appearance', and 'Phonebook'. The 'Dialing' tab is selected. It contains the following settings:

- 'Enable auto-dial by location:' with a checked checkbox and a list box containing 'New Location (the current location)'.
- 'Number of redial attempts:' with a spin box set to 0.
- 'Seconds between redial attempts:' with a spin box set to 15.
- 'Idle seconds before hanging up:' with a spin box set to 300.
- A checked checkbox for 'Redial on link failure'.

At the bottom are 'OK' and 'Cancel' buttons.

**Figure 47 - User Preferences Screen**

These same properties should be set from the More button on the Main Menu/Logon Preferences. **Figure 48** displays the appropriate settings.



The 'Logon Preferences' dialog box has four tabs: 'Dialing', 'Callback', 'Appearance', and 'Phonebook'. The 'Dialing' tab is selected. It contains the following settings:

- A text box stating: 'These preferences apply to 'Login using Dial-Up Networking' at Ctrl-Alt-Del login. You, as an administrator, have access to this sheet.'
- 'Number of redial attempts:' with a spin box set to 0.
- 'Seconds between redial attempts:' with a spin box set to 15.
- 'Idle seconds before hanging up:' with a spin box set to 300.

At the bottom are 'OK' and 'Cancel' buttons.

**Figure 48 - Logon Preferences Screen**



# DHIAP

## RADIUS Installation and Maintenance Guide

---

### **Troubleshooting Tip: Client Disconnect**

#### **Problem:**

During the testing phase of the CiscoSecure RADIUS compliant system, two instances of abrupt dial-in disconnection were noted. The first instance occurred after a twenty minute period of inactivity. The second instance occurred at the home of an offsite testing participant. Approximately ten to fifteen minutes after the initial connection, the participant was disconnected. This happened during periods of activity and inactivity.

#### **Solution:**

The disconnection during a period of inactivity was due to a default setting of "0" in the "idle time" and "session time" section in Group Settings. If "0" is set, a twenty minute default timeout period is activated. Entries for "idle time" and "session time" should be set in seconds. The second disconnection was due to a faulty modem cable.

#### **Analysis:**

In the event that a user is continually being disconnected from the CiscoSecure RADIUS compliant system, the system administrator should always check the Group Settings "idle time" and "session time" for the appropriate entry. These settings should be implemented according to site policy. If the times are correct, examine all OSI Physical Layer components of the dial-in client.

## **6 BACKUP/RESTORE PROCEDURES**

Provided with Windows NT Server hardware is a Compaq 12/24 DAT. Combined with the standard Windows NT backup server software, it provides a safe method for backing up and restoring crucial system files. These files include all log and activity files as well as the CiscoSecure ACS database. By saving this database, all information concerning user, group, administration and system configurations can be restored in the event of a loss. This will ease administrative burden as well as protect vital information. The following information will provide basic procedures for backup/restorations. It is crucial that the system is backed up to another drive or DAT tape. Additionally, the local log files must be deleted from the local drive after they have been backed up. This should be done approximately every seven days. If they are not deleted, the local hard drive will fill up and cause the system to crash. The problem can be corrected by backing up all log files to an external drive or tape drive and deleting them off the local drive. Log File Archive in **Section 6.3** provides greater detail pertaining to this subject.

### **6.1 Backup**

Insert the DAT tape into the tape drive. From the Start menu go to:

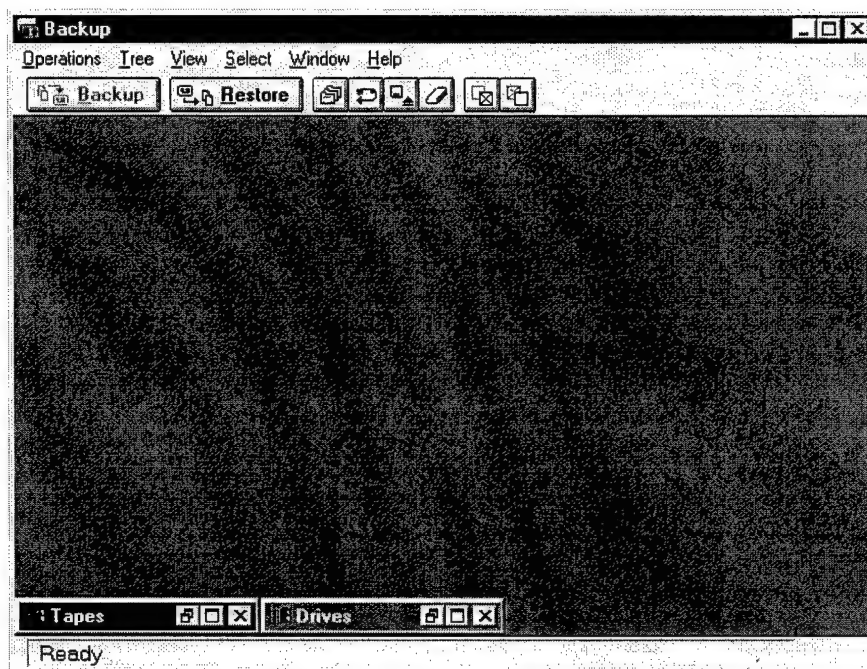
1. Programs
2. Administrative Tools
3. Backup

## DHIAP

### RADIUS Installation and Maintenance Guide

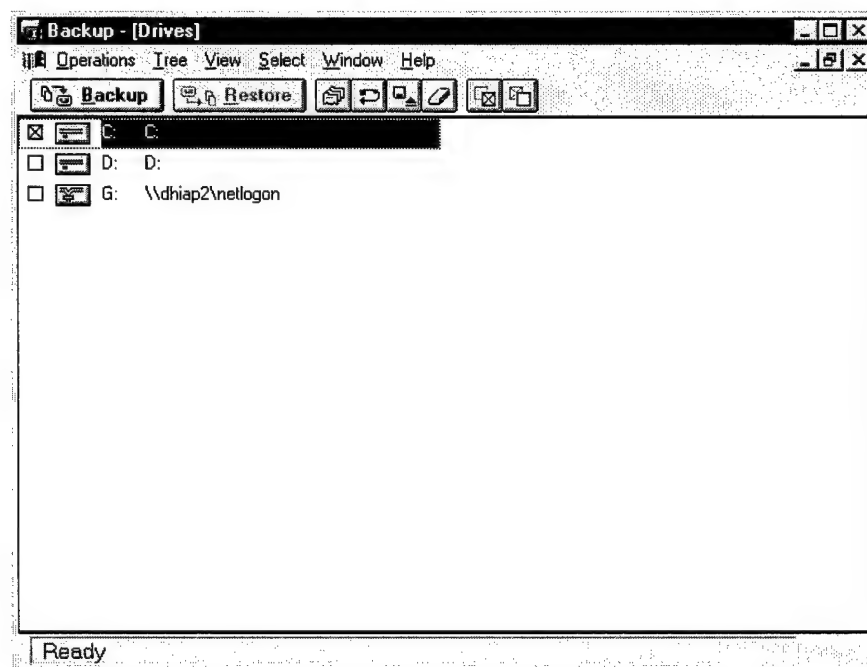
---

The application will open to the screen displayed in **Figure 49**.



**Figure 49 - Initial Backup Screen**

Maximize the Drives window located at the bottom of the window. Once maximized, click on the C:\ drive as depicted in **Figure 50**. By double clicking C:\ individual files may be specifically backed up in place of the entire drive.



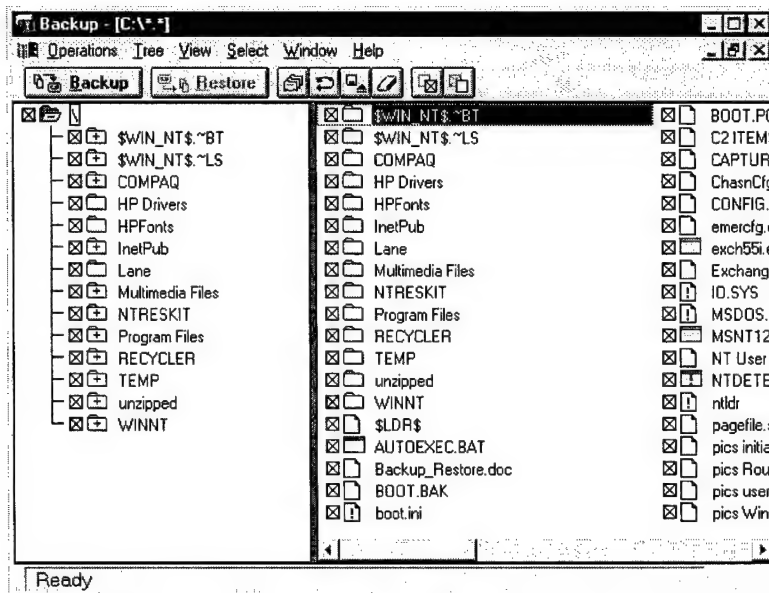
**Figure 50 - Drive Letter Backup Screen**

## DHIAP

### RADIUS Installation and Maintenance Guide

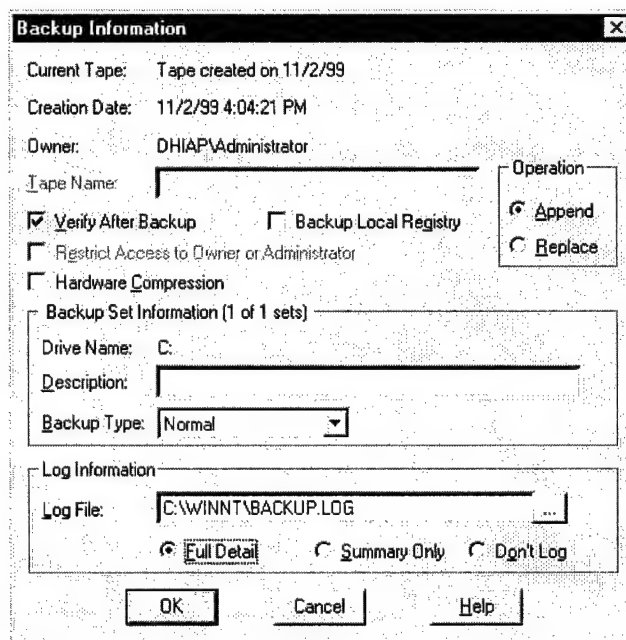
---

**Figure 51** depicts the selection of individual files after double clicking C:\



**Figure 51 - File Selection Backup Screen**

Once the appropriate files have been selected, click the Backup button located at the top of the window. **Figure 52** depicts the window that will open and will allow the administrator to set parameters concerning the backup. The procedure concerning the tape name, appending or replacing data, and verification of backup should be determined by local policy.



**Figure 52 - Backup Information Screen**

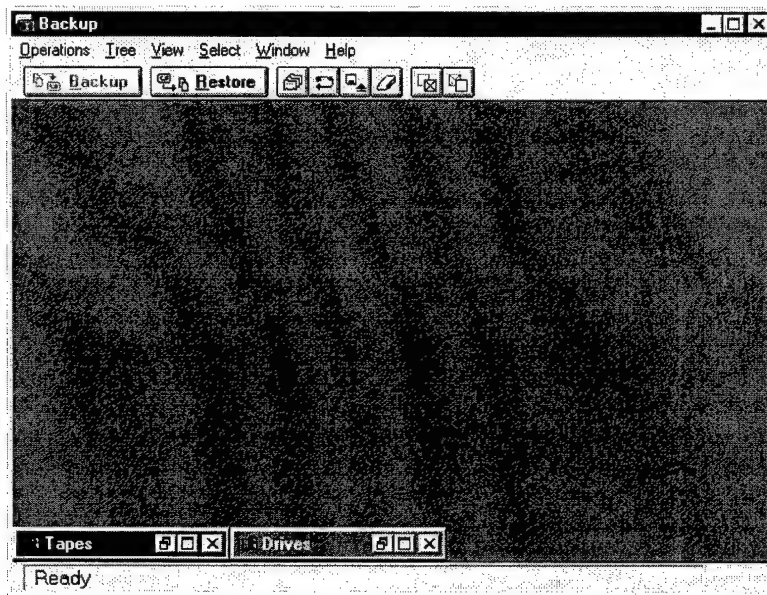
Once the correct information has been selected, click OK and the backup process will start.

# DHIAP

## RADIUS Installation and Maintenance Guide

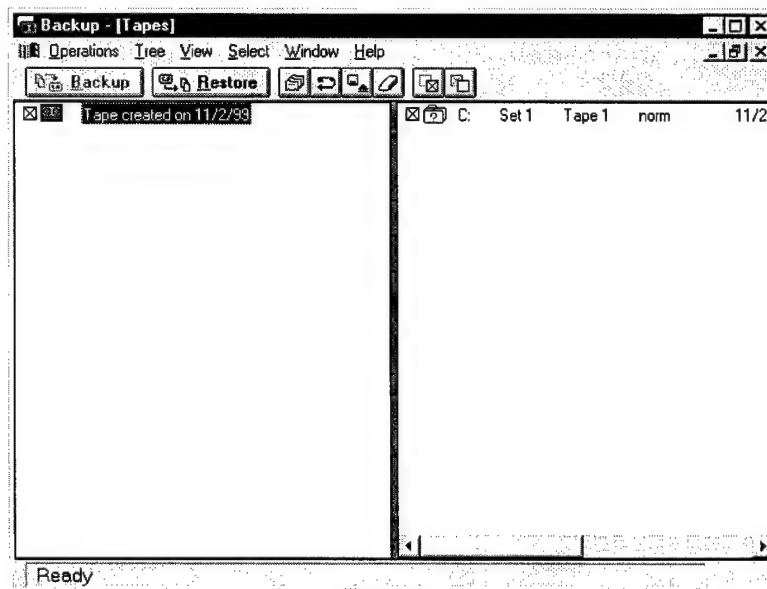
### 6.2 Restore

The restoration process is much like the backup process in reverse. To restore information, place the tape with the desired information in the tape drive. Just as in the backup procedure, go to Programs, Administrative Tools and click on Backup. Once the application has opened, the screen in **Figure 53** will appear.



**Figure 53 - Initial Restore Screen**

Click on the Tapes window to maximize. Once the window is open, click on the tape displayed in the window as in **Figure 54**.



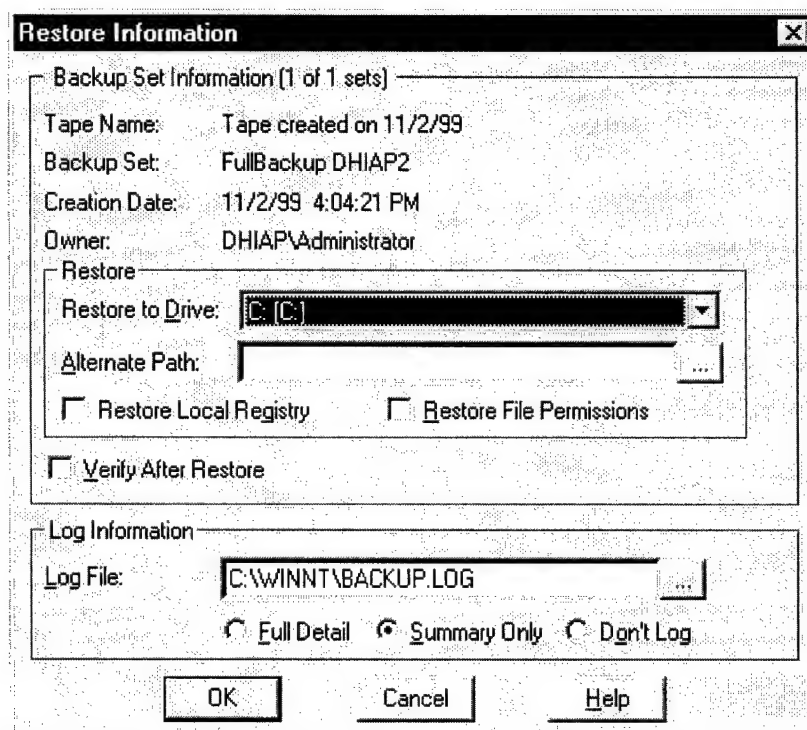
**Figure 54 - Tape Selection Screen**

## DHIAP

### RADIUS Installation and Maintenance Guide

---

If individual files are to be restored in place of the entire tape, double click on the tape and then proceed to select the desired files. Once selected, click the restore button at the top of the screen. The screen in **Figure 55** will appear.



The image shows a Windows-style dialog box titled "Restore Information". It contains two main sections: "Backup Set Information (1 of 1 sets)" and "Log Information".

**Backup Set Information (1 of 1 sets)**

- Tape Name: Tape created on 11/2/99
- Backup Set: FullBackup DHIAP2
- Creation Date: 11/2/99 4:04:21 PM
- Owner: DHIAP\Administrator

**Restore**

- Restore to Drive: C: [C:] (dropdown menu)
- Alternate Path: (text field with browse button "...")
- ☐ Restore Local Registry
- ☐ Restore File Permissions
- ☐ Verify After Restore

**Log Information**

- Log File: C:\WINNT\BACKUP.LOG (text field with browse button "...")
- ☐ Full Detail
- ☒ Summary Only
- ☐ Don't Log

At the bottom are three buttons: OK, Cancel, and Help.

**Figure 55 - Restore Location Screen**

Click OK and the restoration process will start. Once the files have been restored, a message will indicate this. Once the process is complete, the files will be ready for use.

### 6.3 Log File Archive

The CiscoSecure RADIUS compliant system generates many log files documenting system and user events. Information is continually added to the files. The system administrator determines when new files are generated. This can be done on a daily, weekly or a monthly basis. These files are stored on the local hard drive and require a large amount of storage space. Because of this storage requirement, it is necessary to archive the files to another location. If they are not stored elsewhere and deleted from the local hard drive, log file overrun will occur.

Log file overrun is a condition caused when the CiscoSecure software fills up the local hard drive of the RADIUS compliant system with log file entries. Depending on the call-in volume, log file overrun can occur within sixty to ninety days. This renders the overall system inoperable. User authentication will no longer take place and the user groups established in CiscoSecure will no longer be visible to the administrator. This condition can be addressed by implementing the following procedures:

# DHIAP

## RADIUS Installation and Maintenance Guide

---

1. Establish an archive schedule.
2. Configure CiscoSecure to avoid keeping unnecessary TACACS+ and VOIP logs. See **Section 6.3.3**
3. Configure CiscoSecure to automatically delete log files after the administrator has properly archived them to tape for long term (12 month) storage. See **Section 6.3.4**
4. Configure CiscoSecure to delete Debug logs on a monthly basis. See **Section 6.3.5**

### 6.3.1 Archive Schedule

Log files should be archived to tape on a regular basis. They can be archived using most any backup software or the backup utility within Windows NT. Files should be archived at the beginning and end of each month using the "Full" or "Normal" backup option. Each weekday during the month, files should be archived using the "Incremental" option. The "Append" option should be used for each archive so as not to overwrite previous archives. Store and replace the existing tape every thirty days. The following table provides a basic log file archive schedule:

#### Suggested Archive Schedule

<b>Month</b>	M1	M1	M1	M1	M2
<b>Week</b>	W1	W2	W3	W4	W1
<b>Day of Week</b>	M-F	M-F	M-F	M-F	M-F
<b>Backup Type</b>	F,I,I,I,I	I,I,I,I,I	I,I,I,I,I	I,I,I,I,F	F,I,I,I,I

#### Key:

M1 - Month 1

W1 - Week 1

F, I - Full or Incremental backup based on five day work week.

Note: This schedule does not take into account months with thirty-one days nor months starting in the middle of the week. The system administrator should modify it as needed.

### 6.3.2 Types of Log Files

There are two types of files that need managing within the CiscoSecure RADIUS compliant system; Debug log files and Standard log files.

#### 6.3.2.1 Debug Log Files

Debug log files are used to troubleshoot the RADIUS compliant system. They are generated when the administrator logs into Windows NT and one of the specified services is used. All activities pertaining to that service are entered into the debug logs. The system administrator can manage these logs by setting the level of detail, frequency of log file generation and automatic deletion of the logs after a specified number of days. It is usually not necessary to retain these logs. Cisco recommends running a "normal" level of detail until the system has been running for an extended period of time without problems, then, the debug logs can be turned off. Note that the Cisco Systems help desk may ask to run the debug logs at a "high" level of detail to work on system problems. After the system has been corrected, return to the "normal" level of detail or turn off as appropriate.

## DHIAP

### RADIUS Installation and Maintenance Guide

---

The files are stored on the local hard drive in the following locations:

C:\Program Files\CiscoSecureACS V2.4\CSAdmin\Logs  
C:\Program Files\CiscoSecureACS V2.4\CSAuth\Logs  
C:\Program Files\CiscoSecureACS V2.4\CSDBSync\Logs  
C:\Program Files\CiscoSecureACS V2.4\CSLog\Logs  
C:\Program Files\CiscoSecureACS V2.4\CSMon\Logs  
C:\Program Files\CiscoSecureACS V2.4\CSRADIUS\Logs  
C:\Program Files\CiscoSecureACS V2.4\CSRADIUS\Logs

#### 6.3.2.2 Standard Log Files

Standard log files are defined by the system administrator during setup and document information pertaining to user session statistics. These statistics are necessary when troubleshooting the overall system and when investigating suspicious activity. They should be retained in a long-term storage location. By default, standard log files are stored on the local hard drive in the following locations:

C:\Program Files\CiscoSecureACS V2.4\Logs\Failed Attempts  
C:\Program Files\CiscoSecureACS V2.4\Logs\RADIUS Accounting  
C:\Program Files\CiscoSecureACS V2.4\Logs\AdminAudit  
C:\Program Files\CiscoSecureACS V2.4\Logs\Backup and Restore  
C:\Program Files\CiscoSecureACS V2.4\Logs\DBReplicate  
C:\Program Files\CiscoSecureACS V2.4\Logs\DbSync  
C:\Program Files\CiscoSecureACS V2.4\Logs\ServiceMonitoring  
C:\Program Files\CiscoSecureACS V2.4\Logs\TACACS+Accounting  
C:\Program Files\CiscoSecureACS V2.4\Logs\TACACS+Administration  
C:\Program Files\CiscoSecureACS V2.4\Logs\VOIP Accounting

Just as with the Debug log files, the system administrator can manage Standard logs by setting the level of detail, frequency of log file generation and automatic deletion of the logs after a specified number of days.

#### 6.3.3 TACACS+/VOIP Log Files

The files listed below are considered Standard log files, however, the logging feature for these files should have been deactivated during the installation process. Deactivation of these files should be checked and seen in **Section 4.3.2** of these instructions.

C:\Program Files\CiscoSecureACS V2.4\Logs\TACACS+Accounting  
C:\Program Files\CiscoSecureACS V2.4\Logs\TACACS+Administration  
C:\Program Files\CiscoSecureACS V2.4\Logs\VOIP Accounting

#### 6.3.4 Archive and Deletion

Once an Archive schedule has been established and the appropriate files have been identified, Standard log files need to be archived to tape and deleted from the local hard drive. Some of

## DHIAP

### RADIUS Installation and Maintenance Guide

---

these files have additional configuration parameters that can be set. This is discussed as an option in **Section 6.3.4.3**.

#### **6.3.4.1 Standard Log File Archive**

The active Standard log files should be archived to a tape drive by completing the following steps:

1. Place a DAT cartridge in the appropriate drive on the RADIUS compliant system.
2. Start the BACKUP Process as described in **Section 6** of this document.
3. Double click on the C:\ icon as shown in **Figure 50**.
4. Double click the Program Files folder
5. Double click the CiscoSecureACS V2.4 folder.
6. Click the Clear box next to the Logs folder. An X will appear in the box.
7. Click the Backup button at the top of the page.
8. A backup configuration screen will appear.
9. In the Backup Type section, select "Normal" for the first backup or "Incremental" for a mid week backup as needed.
10. Select "Append".
11. In the Description Section, type the current date.
12. Click OK.
13. The backup process will start.

Note: When using the Windows NT Backup Utility, the system administrator must start the archive process manually. When using backup software such as Arc Serve or Backup Exec, the archive process can be automated.

#### **6.3.4.2 Standard Log File Automated Deletion**

Once the Standard log files have been archived to tape, CiscoSecure should be configured to automatically delete the files on the local hard drive. To automatically delete these files, complete the following steps:

1. Open the CiscoSecure administrative software.
2. Click the System Configuration button.
3. Click the Logging hyperlink.
4. From the Log Target menu, click the CSV Failed Attempts hyperlink.
5. Place a check mark in the Manage Directory check box.
6. Click the radio button "Delete files older than 30 days".
7. Click the Submit button.
8. You will automatically be returned to the Log Target menu.
9. Perform steps 4-6 for CSV RADIUS Accounting.
10. Complete this process for CSV Admin Audit starting from the Administrative Control button.

#### **6.3.4.3 Standard Log File Migration**

Two of the Standard log files have the configuration option to be migrated to a network drive as well as archived to a tape. These files are:



## DHIAP

### RADIUS Installation and Maintenance Guide

---

C:\Program Files\CiscoSecureACS V2.4\Logs\Failed Attempts  
C:\Program Files\CiscoSecureACS V2.4\Logs\RADIUS Accounting

To migrate the files listed above to a network drive, complete the following steps:

1. Open the CiscoSecure administrative software.
2. Click the System Configuration button.
3. Click the Logging hyperlink.
4. From the Log Target menu, click the CSV Failed Attempts hyperlink.
5. In the Directory section, type the path to the network storage drive.
6. Click the Submit button.
7. You will automatically be returned to the Log Target menu.
8. Perform steps 4-6 for CSV RADIUS Accounting.

#### 6.3.5 Debug Log Files Deletion

The Debug log files listed in **Section 6.3.2.1** should normally be deleted every thirty days unless directed otherwise. Configure CiscoSecure to automatically delete these files by completing the steps below:

1. Open the CiscoSecure administrative software.
2. Click the System Configuration button.
3. Click the Service Control hyperlink.
4. Place a check mark in the "Manage Directory" check box.
5. Click the radio button "Delete files older than 30 days".
6. Click restart.

**Warning - The procedure above also deletes the following Standard log files:**

C:\Program Files\CiscoSecureACS V2.4\Logs\AdminAudit  
C:\Program Files\CiscoSecureACS V2.4\Logs\DBReplicate  
C:\Program Files\CiscoSecureACS V2.4\Logs\Backup and Restore  
C:\Program Files\CiscoSecureACS V2.4\Logs\DbSync  
C:\Program Files\CiscoSecureACS V2.4\Logs\ServiceMonitoring

To prevent unwanted log file loss, keep the deletion time to thirty days. If an archive of these files has not been done prior to the automatic deletion, files will be lost and unrecoverable.

#### Recommendations

Log file overrun can quickly fill the RADIUS compliant system's hard drive causing the system to crash if not managed properly. By creating an automated process to delete the Service Control log files on a regular basis, the chances of log file overrun can be kept to a minimum. The system can be further enhanced by moving the Standard log files to a network drive or tape. Only a minimum number of files can be migrated to a network drive. To eliminate multiple storage locations, it is highly recommended that all standard log files be archived to tape on a regular basis.

# DHIAP

## RADIUS Installation and Maintenance Guide

---

### 7 UNINTERRUPTIBLE POWER SUPPLY

The CiscoSecure RADIUS compliant systems in the DHIAP demonstration have an Uninterruptible Power Supply (UPS) attached to them which provides a twenty minute backup. The administrative software, PowerChute, for the USP is located on the AAA Server.

#### **Troubleshooting Tip: Uninterruptible Power Supply Software Broadcast**

##### **Problem:**

An Uninterruptible Power Supply (UPS) is attached to every CiscoSecure RADIUS compliant system. While most sites have an UPS for the entire LAN, this one is provided as a twenty minute backup for the CiscoSecure RADIUS compliant system. Minor configuration changes were being made to an onsite system and the UPS had to be disconnected from the Windows NT server. This in turn resulted in a signal loss between the two. The server UPS software sent a message to all nodes on the LAN, indicating that a power failure had occurred. In the event of a power loss that only affects the CiscoSecure RADIUS compliant system, all nodes attached to the LAN will receive this message. This message does not necessarily pertain to the normal user and caused many help desk calls.

##### **Solution:**

The uninterruptible power supply is configured by PowerChute software located on the Windows NT server. This software will send an alert for different events that pertain to power related events. By default, all domain users are to be notified in the event of a signal loss or power fluctuation. User notification policy can be changed from the UPS Configuration menu by clicking Event Actions. The Options tab under Notify Users should be configured to notify only those individuals selected by the system administrator.

### 8 MICROSOFT EXCHANGE

Microsoft Exchange is accessible to all dial-in users. It can be accessed over the World Wide Web or via Microsoft Outlook configured on the dial-in workstation. There are two ways to access a mail server using CiscoSecure. The first method of access is through the World Wide Web. The MS Exchange server must also support this procedure. The second method of access is via a desktop icon on the dial-in workstation.

World Wide Web access to the MS Exchange server is possible from a dial-in user if the terminal is properly configured. If a site is using an Access Control List (ACL) to limit web access, the IP Address of the MS Exchange server must be included as an entry in the list.

An example configuration line in the ACL is:

`access-list 150 permit tcp any host X.X.X.X (IP Address of the MS Exchange Server)`

This entry will permit the dial-in user to access the MS Exchange server via the web.

Access to the MS Exchange server may also take place using the MS Outlook mail program loaded on the dial-in workstation. To activate the program, the end user will click an MS Outlook icon on the workstation desktop. The end user will then have the opportunity to work

## DHIAP

### RADIUS Installation and Maintenance Guide

---

offline to prepare and read their mail as needed. To support this effort, MS Outlook should be loaded onto the dial-in workstation while dialed into the CiscoSecure server. This will ensure that the data path is defined correctly when identifying the MS Exchange server. The program must be set up as it would normally be done on a local area network. However, if an ACL is being used, it must contain information that permits the user to access the appropriate MS Exchange mail server. The ACL must include the traffic type and server IP Address. If mail management and delivery is divided between two or more servers, the IP Address of the bridgehead must be included in the ACL. The following example will display the appropriate entry for a bridgehead. If no bridgehead is used, then enter the IP Address of the MS Exchange mail server.

**Example:**

access-list 150 permit tcp any host Y.Y.Y.Y (**IP Address of the Bridgehead to the MS Exchange Server**)

Once the ACL is implemented and if communications between the dial-in workstation and MS Exchange server can not be established, then the IP Address of each MS Exchange server should be included in the ACL.

**Example:**

access-list 150 permit tcp any host Y.Y.Y.Y (**IP Address of the Bridgehead to the MS Exchange Server**)

access-list 150 permit tcp any host P.P.P.P (**IP Address of the first MS Exchange Server**)

access-list 150 permit tcp any host Z.Z.Z.Z (**IP Address of the second MS Exchange Server**)

---

# **Remote System Administration**

---

## **Issues and Recommendations**



**Authors:** Stephen L. Packard, LMES

Archie D. Andrews, ATI

April 2000

ATI IPT Special Report 00-05

This work is supported by the U.S. Army Medical Research and Materiel Command under Contract No. DAMD 17-99-C-9001. The views, opinions and/or findings contained in this report are those of the authors and should not be construed as an official Department of the Army position, policy or decision unless so designated by other documentation.

This page left intentionally blank.

**BACKGROUND**

A special investigation was performed in DHIAP Phase I to assess the important issue of how to balance the economy of remote central administration of standard information systems with a site's responsibility for the security of its systems. Military Medical Treatment Facilities (MTFs) operate about a dozen "standard" information systems developed by higher echelon agencies and frequently administered by those agencies or their surrogates. At best, that remote administration represents a very large unknown to the local information system security officer (ISSO). At worst, it represents an easily compromised conduit to the remotely administered systems and every other system on the MTF's network. Because the remote administration is performed by or contracted by a higher headquarters (Army Medical Command, OSD/Health Affairs, Regional MTF or Program Manager for CHCS), the MTF often has little knowledge of the identity or location of the administrator and no influence over the tools and techniques used by that administrator.

**RESEARCH OVERVIEW**

The research was based on the hypothesis that effective policy, operational procedures, management support, and monitoring tools would reduce the potential risks created by remote administration. It evaluated remote system administration in terms of actual practices, inter-agency relationships, and existing policies, then developed potential changes in policies and practices to improve MTF control of external administrators, to minimize exposure of administrators' traffic, and to reduce exposure of other local systems should the remotely administered system be compromised. The major activities of the research effort are listed below.

1. Reviewed the results of the DHIAP ISEs. Those results included several instances of remote system administration. Emphasis of the review was on MTF information security policies, controls the MTFs exercise over external administrators, isolation of externally administered systems, and plans for the future.
2. Interviewed the administrators / program managers of those systems that operate within MTFs' networks and are administered from an external location. The objective of these interviews was to ascertain the information assurance measures the administrators observe and to better understand their motivations, constraints, policies, and plans for the future.

3. Investigated existing and emerging technology for protected remote access, for monitoring administrators' activities, and for system isolation.
4. Investigated modification of MTF and higher echelon policies to meet the mission of both with the minimum exposure of sensitive information and systems processing that information.

## **PRESENT MTF OPERATIONAL ENVIRONMENT FOR REMOTE SYSTEM ADMINISTRATION**

### **Reasoning Behind the Army's Use of Remote Administration**

The DHIAP Team evaluated the information security of two medical treatment facilities (MTF). Each site operated computer systems that had been developed by a higher echelon, delivered, and centrally managed. The central management was the logical, economical approach at the time of delivery. Scarce resources make it inefficient for each site to acquire and train staff to manage each of several, heterogeneous systems. The higher echelon could acquire and sustain a small, skilled team to perform system maintenance and administration, user help-desk, and functional enhancement from a remote location. Economic benefits were obvious. Additionally, this approach led to a much higher degree of configuration control than an approach in which the MTFs performed these tasks.

### **Negative Impact of Implementing Advanced Technologies**

Over time, all of the systems at the MTF were interconnected via local area network (LAN) to reduce the number of terminals and workstations needed by each individual and to simplify collection of information from different systems. While networking brought new efficiencies to the MTF, it also blurred the lines of responsibility for information security. The MTF ISSO was responsible for all information and information systems within the MTF, but those remotely managed systems were not under her/his control. Often, the MTF ISSO is not aware of the actions taken by the remote administrator or the mechanisms used by the remote administrator to protect the system while performing those actions.

### **Variability in Conduct of Remote Administration**

Relationships between MTFs and remote administrators vary. In some cases, the remote administrator contacts the MTF prior to any action, requests temporary access, reports all actions performed, and conducts those actions via end-to-end protected session. This relationship is the exception.

Normally, the remote administrator is a contractor tasked by, funded by, and responsible to the higher echelon. His/her only motivation for communication with the MTF ISSO is to ensure the MTF does not interfere with the administration of the given system.

### **Variability in Policy Guidance for Remote Administration**

Information security policy at the MTFs varies widely. The existence of higher echelon policy is both a blessing and a curse. Some MTFs simply assemble higher headquarters policy to satisfy their requirement for a local policy. Others have made the effort to understand what information they are bound to protect, for whom they are protecting it, from whom they are protecting it, when and where it is most vulnerable, and how adversaries might try to compromise it. With this understanding, they are able to select some higher echelon policy "as is," modify or adapt other existing policy, and develop local policy to address the local environment.

### **Prevalence of Remote System Administration**

Because the DHIAP Team had access to only two MTFs, they conducted interviews with the parent organization of one of the participants. This organization is experiencing the same conflict between maximum central control of computers and delegation of computer security responsibility to the lowest level. In this organization, the Chief Information Officer (CIO) uses a central staff and workstations for network monitoring, help-desk support, and standard software deployment. As with remote system administration for MTFs, this is a reasonable, economical approach, and meets the requirements of the CIO. As with the MTFs, however, it removes control of critical people and actions from the people held accountable for the security of the individual systems. The Team's belief is that this is a widespread problem among many computer-using businesses today. Some recommendations are offered below, but must be carefully tailored to each situation.

## **VULNERABILITIES**

When not managed prudently, remote system administration potentially exposes the MTF to:

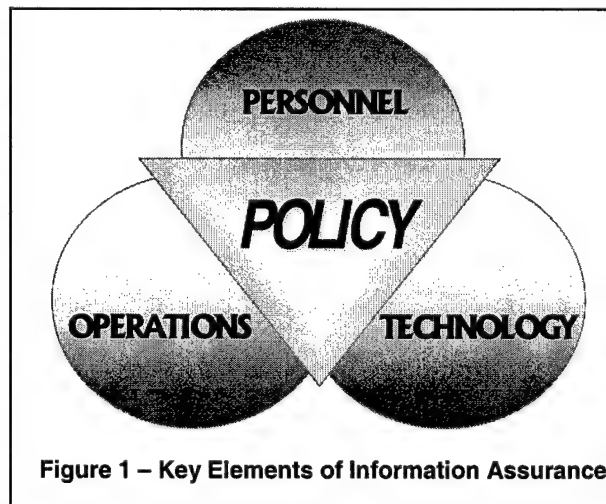
- Interception and compromise of root level passwords
- Interception and compromise of other privileged information
- Unauthorized system/network entry by individuals appearing to be the system administrator (Spoofing)



- Exposure of internal topology to denial of service attack
- Delivery of Trojan Horse or other malicious software
- Administration of a standard system host by unqualified administrators
- Inadvertent alterations to other hosts on the MTF network
- Access to MTF network by administrators who do not meet personnel reliability requirements
- Failure of the remote administrator to comply with local security policies and practices
- Failure of MTF ISSO to comply with Army policy by permitting network access by unknown or unvetted user (the remote administrator)
- Inadvertent or malicious alteration of standard system access controls

## RECOMMENDATIONS

In this paper, recommendations are organized into four areas: Policy, Personnel, Technology, and Operations. As indicated in Figure 1, Policy, Personnel, Operations, and Technology must work together to effect a secured environment. Activity that emphasizes any one area without regard for the others will fail to produce the desired results. For example, acquiring technology to bolster security, but implementing it incorrectly or failing to consider its impact on operations or the personnel who maintain and



administer it, could actually undermine security when it is used improperly or not at all. Similarly, pronouncing new policy without revising procedures, training staff, or enforcing compliance will be a wasted effort because the policy is never implemented. There are numerous examples of well-intended protection measures that have proved ineffective because of failure to consider the interdependency of policy, personnel, operations, and technology.

**Policy Recommendations**

The existence of sound, clear, commander-endorsed policy allows the MTF ISSO to approach the higher echelon sponsor of the remotely administered system and request the remote administrator comply with that policy. Approaching higher echelons without such policy is much less likely to produce the desired, necessary change in remote administrators' behavior.

The DHIAP Team strongly recommends MTF ISSOs take the approach of understanding their local environment and assembling and developing appropriate, meaningful local policy. Sound local policy will be a lasting sub-flooring on the foundation of higher echelon policy. This local policy will be the layer upon which the pillars of technology, procedures, training, and enforcement stand.

The basic elements of the policy regarding remote system administration should be:

- Remote administration is an economic, mission-supportive necessity.
- Remote administration (like any external access) represents a path over which sensitive information flows and through which an attacker could penetrate.
- Remote administrators must comply with the policy for all external access.

The basic elements of the general procedure for all external access would consist of:

- Authenticated and authorized logon - to prevent access by unauthorized users
- Encrypted traffic - to prevent interception and eavesdropping (see Session Protection)
- Audited activities - to monitor activities
- Access frequency and duration limits - to limit use to absolute minimum
- Remote workstation protection - to prevent compromise at remote administrator's point of entry
- All other normal actions to protect the MTF computing environment

**Technology Recommendations**

**Remote Administrators' Protective Measures.** Because of the remote administrator's privileged access, it is critical that communications from his/her workstation to the target system be protected. Vulnerable points include the workstation, any public telephone circuits, any internet service provider servers, any "home office" servers/networks of the higher echelon or the contractor, and any other private or public linkages. For simplicity and consistency, the DHIAP Team recommends protective measures be applied at the workstation and seamlessly from the workstation to the target system. Each of these is straightforward and is supported by numerous commercially available tools and by commonly accepted policies and procedures.

**Workstation Protection.** Only trusted people should have any access to the workstation. This should include physical and virtual access. The workstation should be in a location that is only accessible by employees. If employees other than the administrator(s) have access to the workstation's location, then the workstation should have a protective device. This may be as simple as a keyboard lock or as sophisticated as a biometric device for logon authentication.

In these times of outsourcing and telecommuting, it is conceivable that some remote administration would be performed from private homes. This can be done securely if the workstation is used only for official business and is protected from casual, unofficial use by other members of the household. The risk of not securing the workstation is, of course, the stereotyped adolescent hacker in the household. This person loads a sniffer on the workstation. And returns after a legitimate session having captured all user identification and authentication necessary to establish a secure session, access the host within the MTF network and, perhaps, other hosts on that network.

**Session Protection.** The remote administrator should establish a session that encrypts all traffic from the workstation through all communications media to the target host. There are a number of commercial off the shelf capabilities. Some use secure socket layer (SSL) to encrypt the session from end-to-end, preventing interception of user identification, passwords, or other critical information. Although vulnerable to IP spoofing, some use IP address filtering or lockout. The former is intended to prevent access from any source other than known, trusted sources. IP address lockout is an approach to detect a brute force password attack and lockout the origin of the attack. Still others use virtual private network (VPN) technology.

The following products are listed only to advise the reader that there are many options, and the discussions of these are intentionally very brief because of the short life span of today's technical products. Many of these will have been

overcome by progress before this paper reaches the reader. Some of these products are:

PC Anywhere	Remotely controls Windows 95, 98, and NT computers Supports VPN when VPN-1 Gateway is installed on the target host
LapLink	Remotely controls Windows 95, 98, and NT computers
Remotely Anywhere	Remote administration utility that runs as an NT service. Runs over LAN or through the web. Uses SSL, IP address lockout and filtering. Requires decent browser (Netscape 4.0 or Internet explorer 4.0)
Back Office	Remote administration utility that runs across any TCP connection using a simple console or GUI application.
Cylink	Cylink Corporation produces several tools for end-to-end session protection. Cylink Link Encryptor secures sensitive data transmitted over high-speed, point-to-point communication links. Cylink LSA Encryptor secures remote connections to corporate networks over dial-in links at speeds from 300bps to 56Kbps. Another product deals with frame-relay networks. Cylink Private Wire is well suited for secured remote administration.
F-Secure SSH	Provides end-to-end security in terms of encryption, military-grade encryption and data privacy protection between computers running F-Secure SSH Client and F-Secure SSH Server software.

If communications are via dedicated circuits or public telephone lines from the workstation to the target host, hardware encryption is an option. When available, these devices provide very high degree of protection between government computers.

**IPSec.** One very promising technology is a standard, rather than a product. IPSec is a set of Proposed Internet Standards issued in 1995 by the Internet Engineering Task Force as a specification for Internet Protocol Security. As opposed to a single vendor, proprietary product, IP Sec is being implemented by numerous vendors. IPSec provides three main functions:

- Authentication-only or Authentication Header (AH)
- Encapsulating Security Payload (ESP)

- Key-exchange function

IPSec provides the capability to secure communications across LANs, private and public WANs, and the Internet. This fits very well into the remote administration problem. A virtual private network is established from the administrator's workstation to the administered host and the sessions are protected.

Other products exist and the exact capabilities and vulnerabilities are constantly in a state of improvement. Similarly, individuals who would violate the privacy of others are also constantly improving capabilities of their tools.

### **Operations Recommendations**

**Local Protective Measures.** A basic premise of security of any network is that there are a limited number of possible entry points (one is good) and those are controlled so as to implement the organization's information security policy. The DISC4 policy directing implementation of Remote access dial-up service (RADIUS) is a step in this direction. Although the DISC4 policy only addressed dial-up entries to the local networks, the implementation delivered under the Defense Healthcare Information Assurance Program (DHIAP) lays the groundwork for channeling and authenticating all entries to the local network through a single point. Although intended only to control dial-up authentication and access, emerging extensions to RADIUS may enable the local network administrator a means to channel inbound traffic (internet and MEDNET) through the same controls.

- Enable the account only on request by an authenticated source
- Disable the account at all other times
- Audit all transactions during the period when the account is enabled
- Review audit logs for any suspicious transactions

**Remote Administration Measures.** Following are some measures that the remote administrator can employ to substantially reduce the risk of compromise to an unauthorized person:

- Encrypt remote administration traffic such that attackers monitoring network traffic cannot obtain passwords or inject malicious commands into conversations.

- Use packet filtering to allow remote administration only from a designated set of hosts.
- Maintain this designated set of hosts at a higher degree of security than normal hosts.
- Do not use packet filtering as a replacement for encryption since attackers can spoof Internet Protocol (IP) addresses. (With IP spoofing, an attacker lies about his location by sending messages with an IP address other than his own.)
- Installation of remote administration software on Web servers is a dangerous practice and should be minimized or eliminated. Where this practice is necessary or mandated, the above measures will reduce risk of compromise.

**Personnel Recommendations**

**Remote Administrator Vetting.** Generally, none of the personnel information that an ISSO requires for local users is available regarding remote administrators. A local policy that requires local users to comply with a reasonable level of initial and continuous scrutiny will give the MTF a basis for demanding higher echelons provide this level of information on remote administrators. At the very least, an information security officer or program manager at the higher echelon should assume personal responsibility for the reliability of his/her organization's remote administrator(s).

**Initial User Vetting.** Within a single MTF, the ISSO requires that every user be vetted, that is, positively identified. This normally includes proof that a user is in fact who he/ she claims to be, a certification by the user's supervisor confirming the user's organization and system access requirements.

**Continuous User Reliability Assessment.** Procedural links to the military and civilian personnel systems ensure that the ISSO is advised whenever a user's status changes. The most common change is transfer out of the MTF. The ISSO should remove the user from the access lists upon transfer.

The ISSO would also be interested in changes that might affect a user's reliability. Although the ISSO at an MTF is not protecting classified material, factors such as marital problems, financial difficulty, legal action, and drug problems all impact the user's vulnerability to compromise by a determined outsider. At the very least, the ISSO should contact the user's supervisor and together they should make a conscious decision to continue, terminate, or modify the user's access to MTF systems.

**SUMMARY**

The most effective approach to secure remote administration will be a combination of reasonably cautious operations, appropriate application of technology, and well-placed trust in users, united under a logical, threat-based, locally appropriate policy.

There is no single, magical technological solution that will ensure secure remote administration over a long period of time. Technology changes very rapidly for both the attackers and the defenders of a computing environment. Any of the products practices mentioned earlier will reduce vulnerability of the local network and systems, but only if applied under sound policy and operated conscientiously by trustworthy personnel. Technical staff will need to select products that fit the technical environment then monitor the ever-changing array of products to stay abreast of the equally ever-changing threat.

Unlike purely technical solutions, sound operational procedures and careful assignment of access to users will serve ISSOs well for the foreseeable future. Treatment of remote administrators with the same standards applied to local system administrators in terms of personnel reliability and operational practices, will also be a vital pillar of an effective overall approach.

Developing a sound security policy that is applicable to a broad front is a fundamental first step in addressing the security risk imposed by remote administrators.

---

# Public Key Infrastructure

---

## Resources, Requirements and Recommendations



Author: Kibbee D. Streetman, LMES

Contributors: Archie D. Andrews, ATI

Stephen L. Packard, LMES

April 2000

ATI IPT Special Report 00-06

This work is supported by the U.S. Army Medical Research and Materiel Command under Contract No. DAMD 17-99-C-9001. The views, opinions and/or findings contained in this report are those of the authors and should not be construed as an official Department of the Army position, policy or decision unless so designated by other documentation.



**BACKGROUND**

Public Key Infrastructure (PKI) provides a facility to encrypt and thereby protect the integrity and confidentiality of sensitive electronic information. In addition, it provides a facility for a non-reputable electronic signature. The military has two high-visibility reasons at this time for evaluating PKI and considering its use. First, DoD has announced a strategic direction to take advantage of this capability. Second, the Health Insurance Portability and Accountability Act of 1996 (HIPAA), has established new requirements for protecting the privacy of individually identifiable health information maintained or transmitted in electronic form. The emerging HIPAA standards, which will pertain to all health plans, and healthcare clearinghouses, and most healthcare providers, will be equally applicable to military and civilian healthcare environments.

The PKI technology described in this paper could be an important component in reaching compliance with HIPAA regulations, again in both the military and the civilian worlds. Because HIPAA is new, and because PKI is being assessed and enhanced in both arenas, there is strong reason for military healthcare to monitor and even participate in current activities occurring in the civilian world. This report discusses the technology of public key cryptography and the PKI necessary to implement it and the beneficial functions that can be provided to the military Medical Treatment Facility (MTF). The report outlines the current state of PKIs, the issues to be addressed in deployment of one in general and issues and impacts that need to be considered by the MTF in particular. It will include recommendations for consideration by MTFs and by the DoD medical community in general.

As a component of the Federal government and the DoD, MTFs have a need to develop a PKI implementation strategy in order to facilitate electronic payments, transfer medical records securely, and engage in electronic commerce with insurance companies, other healthcare institutions, and suppliers. Because of the number of PKI initiatives within the Federal government, and the DoD in particular, as well as the large number of suppliers of PKI solutions, the appropriate approach for the medical community needs to be defined.

PKI is an emerging technology, that is continually being improved and the implications of implementing and applying PKI are not well understood. This research will focus on understanding the implications of implementing PKI in a military Medical Treatment Facility (MTF) environment.

**RESEARCH OVERVIEW**

The research for this effort examined the implications of implementing a PKI in a military MTF environment. First, it defined the components of a typical PKI and the security requirements for those components (see Appendix A). Then current implementations in both the government and private industry were studied (see Appendix B). Finally, the general issues to be considered in PKI deployment as well as those peculiar to the MTF environment were defined. The impact of PKI deployment was assessed and recommendations for initiation of a PKI pilot program in the DoD medial arena was developed.

Results of specific research subjects are reported in the remainder of this section.

This paper introduces PKC and PKI technology as a foundation for further undersanding for the interested reader. Appendix A exemplifies and expands on this introduction. Appendix B provides details of PKI implementations in both government and industry. After understanding the operation of PKC and PKI, the impact of implementation can be better appreciated. Examples of Alternatives to PKI are explored and recommendations are provided. The report summary not only condenses the key information presented herein, but also poses unanswered questions for consideration.

**INTRODUCTION TO PUBLIC KEY CRYPTOGRAPHY AND PUBLIC KEY INFRASTRUCTURE**

Public key cryptography (PKC) or asymmetric cryptography uses two keys; one for encryption that is made public and one for decryption that is kept secret. Conventional cryptography on the other hand uses a single or symmetric key for both encryption and decryption. This key must be securely distributed to all users and protected from disclosure. In addition to easier key distribution PKC has another advantage. If  $n$  users wish to employ symmetric key cryptography, they must distribute  $n(n-1)/2$  secret keys. PKC on the other hand only requires the distribution of  $n$  keys which can be made public. PKC can also be used to provide strong authentication, data integrity, and implement digital signatures.

Although public key cryptography was first proposed in 1976 and has a number of advantages over other cryptographic methods, it has had very little implementation in the public sector until recently. For the last twenty years there has been only a small number of applications based on the Rivest, Shamir, Adleman (RSA) algorithm and an e-mail application developed by

Phillip Zimmerman called Pretty Good Privacy (PGP) which used RSA. Because PKC uses calculations based on very large prime numbers (greater than 200 digits long) it is slow compared to symmetric algorithms like the Data Encryption Standard (DES) even in a hardware implementation. Consequently, the approach that has been taken is to use PKC for key distribution and symmetric algorithms for encryption and decryption thus solving both the speed problem and the need for secure key distribution.

The biggest problem associated with using PKC in practical applications is in trusting a public key that is reputed to belong to an individual. Early implementation of PGP dealt with this problem by having people who knew each other sign each other's respective public keys. Using this technique of introductions, it was possible to build a web of trust among the PGP users in a relatively small community. This informal and essentially horizontal framework has several disadvantages. First of all there is the problem of extensibility. While this process can work in a small community of people who know each other, it will not work as the size of the community becomes larger and more geographically dispersed. The second problem, and a more serious one, is the matter of the integrity of the public keys. With this approach there is no effective way for a user to revoke a compromised key or to distribute a new one.

Diffie and Hellman, the developers of PKC, pointed out these problems of verifying that a public key actually belonged to an entity. They proposed using secure online repositories with databases that specified name to key bindings. Such an approach would potentially have throughput problems when many users tried to access the key servers. In 1978 Kohnfelder proposed a different solution that would avoid this bottleneck. He suggested that a trusted entity called the Certificate Authority (CA) vouch for the binding of the public key to its owner. A digital signature from the CA would be used to bind the public key to its holder. This digital signature would be formed by encrypting the public key and owner's identification with the CA's private key to form a digital certificate. Since the public key of the CA would be known to all, anyone could locally verify the name and key binding of any presented certificate by applying the public key of the CA.

A public key infrastructure (PKI) is the key management infrastructure that supports the generation and distribution of public keys and their associated digital certificates. A practical PKI must include these and a number of other services by means of an interoperating system of hardware and software. These services include:

- Reviewing certificate requests through registration authorities (RAs);
- Issuing and revoking digital certificates through a CA;

- Providing storage and retrieval of certificates in a directory;
- Managing encryption key and certificate lifecycles;
- Handling key backup and recovery;
- Providing time-stamping services;
- Providing cross-certification between different organizations with separate CAs; and
- Facilitating authenticated access to PKI-enabled applications in client-server networks.

Successful deployment of PKC technologies and their use in all forms of electronic commerce depends upon the development and implementation of a practical and technically sound PKI that is standards-based. This fact has been recognized for some time and the number of commercial entities that provide some form of PKI or components to support it has grown from several to nearly three dozen. In 1998 the revenue for PKI products and services was around \$100 million and is projected to grow to approximately \$1.1 billion by the year 2003. A listing of current PKI vendors is given in Appendix D.

The government has also been moving to develop a Federal PKI to address the needs of both the civilian and military sectors. There are currently a number of PKI initiatives, working groups, pilot implementations, and projects underway with the goal of realizing a Federal PKI. Within the government there are a number of agencies that have PKI projects specific to their own needs. Components of the Department of Defense (DoD), the IRS, the Defense Information System Agency (DISA), the General Services Administration (GSA), and the Postal Service (USPS) are but a few of the federal entities that have PKI projects underway. Appendix B provides an overview of current PKI implementations in government and private industry.

## **Implementation impact**

### **Burden to MTF of Implementing PKI Tools and Techniques**

Implementation of a PKI for the MTF environment will require careful planning to address a number of issues common to both the MTF and any other enterprise that wishes to use PKC. Since the MTF and other enterprises such as insurance companies and service providers must be able to securely exchange information, they must work together to address these common

issues. The following section discusses some of these issues and defines their impact on the MTF.

**Complexity.** A full scale PKI has an overwhelming number of components. Some are manpower intensive and others involve software and hardware systems. These components cannot be incorporated in a piece-meal manner.

Implementing a PKI for healthcare will require that additional hardware and software be purchased and installed at the MTF. Because the MTF must be able to interoperate with a variety of other entities both internal and external to the military community, it will be important to proceed with careful planning and coordination in order to select the right components and install them in a complete implementation.

**Costs.** Cost estimates for deployment of a PKI depend upon how the enterprise intends to use it. In general, an organization with multiple sites that wants to issue certificates to authenticate access and provide security to e-mail and web applications can expect to pay between \$110 - \$150 per user per year for a 5,000 user implementation. For a 20,000 user scenario the cost could drop to between \$79 - \$124 per user per year based on costs from two of the largest suppliers of PKI systems.

For the MTF improvements in services, enhanced security, and compliance with healthcare regulations should be the business drivers for deployment of a PKI.

**Lack of PKI-Ready Applications.** A major concern is the fact that most of the commonly used applications do not interpret or deal with digital certificates. Some of the major software vendors have begun to address this issue and some applications do support PKI. There are a number of PKI vendors that offer application integration services but changing software in a large enterprise could result in another significant cost.

For the MTF this is certainly an issue. Except for some e-mail and web applications, it is not likely that the MTF can readily use digital certificates with the applications in daily use without modifications to a number of legacy systems. MTF healthcare services depend on a large number of legacy systems. It would be prohibitively expensive to modify each one for PKI. A more cost effective approach is to develop a "front-end" application that provides a PKI function and then directs the legacy system to perform its service for the PKI authenticated user. Several of the PKI vendors listed in Appendix D have products that may be able to provide this service.

**User Acceptance.** One of the values of PKI is that the same type of certificates can be used to authenticate both people and devices. To do this,

client workstations will have to use PKI-ready applications or new software. This could also result in extra steps for the end user to perform as well as extra authentication steps to use a smart card or a biometric.

User acceptance of the PKI and adherence to its security policies is critical to the security of the MTF information systems. No matter how good the security technology, if the users do not follow policies or have sloppy security practices, the systems may be subject to attack and compromise from both the inside and outside. Security practices are not normally a primary concern of the healthcare professional. For this reason it will be important for the MTF to provide user security awareness training for its staff. Security policies should be stated clearly, well-understood and easy to implement in order to ensure good user acceptance.

**Scalability.** Many of the initial deployments and pilot projects for PKI have involved a small number of users. Most organizations will ultimately require a PKI that can grow to hundreds of thousands or to millions of users. This will require more network bandwidth, more directory space and will push the performance of revocation and certificate management tools. Manpower needs will also have to be addressed as a PKI enterprise grows.

The CA and directory capability that the MTF deploys must be able to sustain growth in the user community as more users and applications are added. Likewise, the expansion of trading partners will require a PKI that has the necessary capacity for growth without severe restructuring.

**Emerging Standards.** The Internet Engineering Task Force's PKIX working group has been working hard to develop standards for the different technical aspects of an X-509-based PKI. NIST is developing security standards and PKC vendors have recently made modifications. A number of standards do exist but have been interpreted differently resulting in some confusion in the industry. Interoperability between different standards bodies will also have to be addressed.

Standards are becoming more clearly defined and implemented in the PKI products that vendors are producing. The MTF community should not delay decisions about PKI deployment until all aspects of the X-509 and related standards are resolved.

**CA Interoperability and Cross Certification.** Since an enterprise may have certificates from multiple vendors whether or not they will interoperate can be a real problem. Applications must be able to accept certificates from multiple sources and cross certification of the various CAs that an enterprise must interface with must be possible.

The MTF community must exchange information with a diverse community both inside and outside of the military. Because of this the MTF CA must be able to interact with a number of other CAs. In order to interoperate these various CAs, they must have common security policies that are sensitive to the needs of the total healthcare community, conform to all the existing healthcare regulations, implement high quality key management, and provide for the privacy rights of patients.

**Certificate Revocation.** Revocation of certificates within the internal PKI of an enterprise with a single CA is not a substantial problem. But in an electronic commerce scenario where there are multiple heterogeneous CAs, a real-time revocation capability is virtually impossible today. Because trust in the validity of a certificate and the key (individual) it represents is vital, solving this problem is critical to a widespread deployment of PKC and digital signatures.

Since all certificates, including the CA root, have a period of validity, a related problem is how to reissue certificates when the CA certificate expires. Also, some mechanism must be in place to reissue or renew expired user certificates.

The MTF must not only be able to revoke certification within its own domain but also to receive revocation lists from all the other CAs with which it operates. This again emphasizes the need for the CA interoperability and cross certification mentioned above.

**Key Escrow, Recovery, and Archive.** Key escrow requires that one's private (decrypting or signing) key be held by an outside party so that encrypted files could be recovered if needed for legal or operational reasons. Aside from privacy issues, this concept has had much opposition because the escrow agent can masquerade as the key owner and sign in their place. Because there may be a valid need to recover encrypted data, it has been suggested that two pairs of keys should be used. One public/private pair could be used exclusively for signing and not released to anyone. The second public/private pair would be used for encryption and the private one escrowed to a trusted agent. This approach, however, complicates the PKI deployment because now each user/entity must have two certificates.

In addition to key escrow and recovery, the MTF may need to provide for archiving of keys. Military personnel are subject to reassignments in other areas, ancillary healthcare providers change, insurance and benefits payment plans are modified, all resulting in a need to archive keys that were previously used. This could require the MTF PKI to provide extra storage capacity and recovery procedures for these key certificates.



**Effect of PKI on IM Staff Size and Skills at an MTF.** Implementation of a PKI within the information system of a military MTF will have an impact on the staffing requirements needed for support. The size of this staffing impact will of course depend on the number of people the PKI will support, the type and number of applications that will use the PKI, and the depth of the infrastructure. Another factor is how much of the infrastructure will be managed by DoD. At the very least the individual MTFs will have to manage their own local certificate and directory services.

Because of their importance, a full-time person should be responsible for maintaining the certificate and directory servers. Several additional staff may be required to handle the administrative activities associated with certificate requests and issuance. If external entities are part of the MTF PKI, then the function of an RA will be required. Again, depending on the size of the user community, additional staff resources may be required to install software for end users, to instruct the end users on how to generate and protect their keys and provide security awareness training. A full-time help desk capability should also be provided to assist end users and to help resolve problems.

This group of personnel position types will have to be staffed at a level commensurate with the size of the MTF and its user community.

#### **Review of DoD Pilot PKI Program Based on Netscape's Certificate Management System 4.1**

The DoD has contracted with Netscape Communications Corporation for software to use in future PKI plans and will extend this acquisition to other vendors to also supply the software for digital certificate management authorities. Both efforts are part of the DoD plan to develop an overall PKI to secure its information networks and to move to a paperless contracting environment.

The Netscape contract includes version 4.1 of the Certificate Management System (CMS) and will provide the DoD with the capability to issue and manage digital certificates. These certificates will provide the PKI technology support to authenticate users of network resources, encrypt information flowing over them and support digital signatures and, if successful, will result in the largest single PKI in the world.

The DoD is already using PKI technology in a number of pilot programs including the Defense Travel System and is planning additional ones for electronic commerce and command and control systems applications.



A number of the services are now using certificates from this contract for secure web server access. Further details on the DoD PKI efforts and the follow-on Interim External Certificate Authority program are given in Appendix B.

### **Profile of Typical MTF re: PKI**

#### **Use of PKI With Current Information Systems**

Some current information systems in the MTF may be able to use a PKI capability with little or no major changes. For example, secure web services requires the installation of user and server certificates and server software. However, this does not mean there is no infrastructure needed. Security policies are still needed as well as a means to manage certificates. Some of the vendors given in Appendix D do offer packages that can provide these functions.

Another area that may lend itself to a PKC/PKI implementation is e-mail. The Secure Multipurpose Mail Extension S/MIME protocol as specified in the IETF documents RFC 2630 and 2632-2634 is a method accepted by HCFA and has been adopted by a number of large healthcare institutions for secure e-mail.

Electronic data interchange (EDI) is another area where the MTF can implement S/MIME to exchange data with external trading partners. Several vendors including Cyclone Software and Harbinger have systems and services for EDI and electronic commerce payment systems.

#### **Use of Electronic Signature with Current Procedures**

Electronic signatures can be produced by digitizing a signature by means of a special pad and pen or by cryptographic means to form a digital signature. Software exists for the digital signatures that can integrate signatures into electronic forms and documents and even into workflow systems. The digital signature can be produced by hashing the document or files to be signed using the federal standard specified in FIPS-PUB 180-1 or a commercial algorithm such as MD5. The hash is then encrypted by some public key algorithm using the secret key of the signer as specified in FIPS-PUB 186-1. While there may be instances where digitized signatures are desirable, digital signatures are more widely used and are likely to be the best approach for the MTF.

As noted in the previous section, e-mail is a function that can use digital signatures for authentication of the originator and integrity of the content. In the MTF any message, document, file, or form that exists in a digital format

can be signed and sealed by means of a digital signature. Typical examples could be orders regarding patient treatments, diagnoses, lab test results, medications, insurance forms, and patient records.

Digital signatures as specified by FIPS-PUB 186-1 are recognized as legal and binding by the federal government. Many states have adopted similar digital signature statutes and recognize their legality.

### **Potential Benefit of Using PKI**

The benefits of PKC that can be realized through the use of a robust PKI have been stated a number of times already. The chief benefit to the medical community is that it allows the MTFs to take advantage of a technology that can improve patient care and reduce costs over the long term and most importantly, do so in compliance with HIPAA and other regulations. Such compliance ensures patient privacy and addresses liability issues.

The ability to send and retrieve patient records, share them with specialists for consultations, and to do so in a timely manner from virtually any location all depend on being able to use the communication resources of networks such as the Internet. The barriers to being able to do this are the need to authenticate all parties, to control access to authorized individuals, and to provide confidentiality services to protect patient privacy. An appropriately implemented PKI has the potential for overcoming these barriers and making information technologies an effective tool in healthcare.

### **Resources Required to Implement PKI in MTF Domain**

The resources needed to implement a PKI for a MTF domain depends on a number of factors such as the size of the user community, and the applications that will use the PKI.

In general, the following components may be required:

- Certificate server to securely store user certificates. The server should have a back-up.
- X.500 or LDAP directory server. The directory server should be the only entity that can access the certificate server. It serves as an interface to the users and the external world.
- X.509 v3 certificates for the user community.
- SSL-enabled web server and browser software if that capability is to be provided.

- Staff to administer the CA and its policies, enroll users, install user software, and provide user support. End user support may be the most costly aspect of a PKI aside from hardware.
- Optional tokens or smart cards to store user credentials.

### **Alternatives to PKI, Pros and Cons**

The important functions provided by PKC namely, strong authentication, data integrity, non-repudiation, and confidentiality can be provided without a formal PKI. As noted in the introduction, there are a number of applications that can be used to provide secure e-mail and digital signatures.

PGP for example has been integrated into a number of common e-mail applications. There are about eight PGP key servers worldwide that are more or less synchronized to provide key directory services. Using this approach, a doctor in a military MTF can generate a key pair and send the public key to these servers. If a sufficient informal network is build up, then users can send e-mail or e-mail with attachments that have all of the desired PKC protections. The one thing that this approach does not address is trust in the public keys. For a small user group that is co-located even that problem can be overcome. However, in a diverse, extended user community, this approach may not be practical.

Another example of PKC functionality without a formal PKI is the approach taken by Catholic Healthcare West, a not-for-profit healthcare organization in the western United States with more than 12,000 end-users. Catholic Healthcare West is utilizing a server-based Secure MIME (S/MIME) encryption and digital signature functionality provided by a Worldtalk WorldSecure™ Server for seamless and transparent encryption of documents and sensitive information exchanged over Internet e-mail. By ensuring the confidentiality of patient-related medical data over electronic communications, Catholic Healthcare West will be able to comply with HIPAA (Health Insurance Portability and Accountability Act of 1996) and other federal security regulations and reduce its liability risks.

Using Secure MIME (S/MIME), the de facto industry standard for secure e-mail, Catholic Healthcare West has established secure e-mail links over the Internet with its 48 hospitals, healthcare providers, medical labs and other trading partners. This growing S/MIME Network allows Catholic Healthcare West to communicate time-sensitive data quickly and securely. This simple and cost-effective approach to secure e-mail has allowed Catholic Healthcare West full enterprise deployment without the need to install and maintain an extensive Public Key Infrastructure (PKI)

There are other approaches such as using a web-based system like Netscape and the Secure Socket Layer (SSL) protocol to establish secure web browser sessions across the Internet. St. Joseph's Hospital in Marshfield, Wisconsin, is using this technique to allow physicians to securely retrieve patient records from the hospital's mainframe computer using a commercial software system that provides certificates and integrates with their patient records system.

While successful in the instances such a piece-meal approach does not lend itself to the Military Health System (MHS) and its MTF components. Because the MHS must function within a tri-service and managed healthcare environment and must be responsive to the public law, the approach to healthcare information security must involve a comprehensive information protection program supported by common PKI.

## RECOMMENDATIONS

There is a need within MTF information systems for user identification and authentication and access controls based on them. When information is sent outside of the area controlled by the MTF, federal regulations require that privacy or confidentiality controls are used to protect that information from unauthorized access. Electronic forms, payments, and procurements will require a means for providing an electronic signature to ensure validity. Public key cryptography (PKC) has the potential for providing all of these needed functions. A public key infrastructure (PKI) must be developed to securely manage the encryption keys and to revoke expired or compromised ones.

Implementation of a PKI requires thorough planning and attention to the issues that have been discussed in this paper. Since some of the major healthcare institutions in the private sector as well as a number of entities in the federal government and the DoD in particular are developing PKIs, the DoD medical community should begin planning in order to move in a similar direction. The following are some suggested steps that could be taken for an initial phase of PKI development and PKC integration at the MTF.

- *Define an application where the use of PKC is appropriate and provides added value.* Since electronic mail is one of the most pervasive applications, it should serve as a likely candidate. This would provide user authentication, non-repudiation, message integrity, and confidentiality. Files, in the form of attachments, could be transferred securely.
- *Develop the policies that will govern the use of PKC technology and guide the PKI implementation.* The certificate policy should reflect both the business needs of the MTF and the appropriate security requirements to be met. For a healthcare institution such as the military MTF, policies must

reflect the requirements necessary to comply with healthcare licensing, accreditation, and regulations. Certificates issued under MTF CA policies must support the granularity of the subscriber community and be tied to the issuer's obligations under HIPAA and other healthcare regulations. Development of policies will serve to guide the MTF in the selection of an appropriate PKC technology solution.

- *Choose a source for public key certificates.* While the MTF could act as its own certificate authority (CA), for an initial phase implementation it might be more cost effective to use a third-party CA service such as one provided by the General Services Administration's ACES program. A number of other sources are available within the DoD Interim External Certificate Authority program and their use may be required.
- *Investigate the use/integration of smart cards into the PKI.* Since the DoD is requiring the use of smart-ID cards for network access controls, it may be possible to use them to store users' certificates. This would have the advantage of allowing users to operate from workstations other than the one that normally stores their PKI credentials. This would also support remote access security.
- *Evaluate COTS products that provide integrated certificate services.* Appendix D of this paper provides a list of all the current vendors of PKI products and services. Among these are a number that have products that can be used to integrate PKI into existing applications. Examples include Entrust/PKI version 5.0 and Baltimore Technologies UniCert. Silaris Technology has a product, ApprovIt, that implements digital signatures into the document workflow process and allows for multiple signatures on routed documents.
- *Begin the process of studying the integration of PKI into a major MTF application such as CHCS.* To realize the full potential of PKI will require its integration into the applications that are most widely used. If CHCS is going to be upgraded it would be a good opportunity to incorporate PKI support into it.

## SUMMARY

Public key cryptography (PKC) offers a number of critical information security functions. In addition to privacy, PKC can be used for enhanced user identification and authentication, data integrity protection, and non-repudiation through digital signatures. All of these functions have application in Medical Treatment Facilities (MTF) for achieving information security

required by DoD policies and existing and pending healthcare regulations such as the Healthcare Insurance Portability and Accountability Act of 1996.

As more medical files and records are developed in electronic digital form and electronic commerce including claims and payments develops, the need for the functionality provided by PKC will become critical to the operation of MTFs. However, for the full potential of this technology to be realized, there must be an infrastructure developed that can provide trustworthy encryption key management. Such a public key infrastructure (PKI) must be capable of securely generating, distributing, and revoking users' public keys and be interoperable with the PKI of other institutions and entities.

Achievement of an appropriate PKI requires answering a number of critical implementation questions, such as what PKI architecture to use, how user registration is to be done, and which applications are to be enabled to use PKC. For applications and processes that are already automated in the MTF environment, authentication and confidentiality can be provided through use of a PKI. There are currently six production PKI programs running in the federal government and more than a dozen significant pilot efforts that could be utilized by the MTF community.

Medical Treatment Facilities (MTFs) will have to deal with a number of issues common to all entities that engage in electronic commerce or interface with financial organizations such as health insurance companies and their counterparts in the federal government. Military healthcare will deal with civilian health infrastructure in certain areas in the future as a variety of military health responsibilities are more closely coupled to or completely outsourced to civilian suppliers. It seems inevitable that military health will have to exchange information with civilian organizations including:

- The insurance companies for electronic claims submission, payment, etc.;
- HMOs and civilian hospitals where a military patient is treated or who has prior medical documentation about a patient being treated by the military;
- Physician group offices and individual physicians who practice concurrently in the military and civilian worlds; and
- Suppliers of medical equipment/supplier if this falls outside the way they treat DoD suppliers in general.

MTFs may be further constrained by the directives and approaches that are imposed by their command structures and the DoD. Within the DoD and other elements of the Federal government there are a number of programs, initiatives, and pilots for deploying and supporting a PKI. These will have an

effect on the direction that the military MTFs follow or, at the very least, provide a number of options from which to choose.

The following issues will need to be addressed:

- *What to use PKC/PKI to support.* Will the PKI be used primarily for authentication, digital signature, or physical or logical access controls? What applications will need the functionality PKC/PKI can provide?
- *Composition and size of the user community.* The PKI will need to support a diverse group of users (subjects) that are part of the MTF or interact with it including licensed healthcare professionals, persons affiliated with licensed healthcare organizations, healthcare organizations that qualify as a payer or provider, agents for healthcare organizations, and patients for their use in communicating with the MTF concerning their health information.
- *Budget and staffing requirements.* Depending upon the approach taken, a PKI for MTFs can involve significant costs and require a sizeable staff to maintain and support it. Even if the PKI solution is imposed from above and funded it will still require a support staff and a budget for maintenance. Who will provide needed funds?
- *Integration of PKC into existing applications.* MTFs across the services typically use almost 100 custom software products to process medical data and interface with other healthcare entities. None of these applications can use PKC without modification. For applications like the Composite Health Care System (CHCS II) this could require a major modification.
- *Certificate maintenance.* Military personnel frequently move to different assignments. It will be a substantial problem to maintain or re-issue certificates for them unless some form of token such as a smart card is used to carry their credentials. Even then cross-certification between CAs may still be a problem.
- *Smart card infrastructure needed.* The DoD and GSA are planning to use a smart card-based common access card for identification and access to physical facilities and computer networks. If this card can also carry public key certificates, it could be part of the PKI. An infrastructure of readers and software will be required. Use of the smart card ID may be directed by DoD policy.
- *Interoperability issues.* Interoperability with other MTFs, other government agencies, insurance companies, private healthcare providers, and suppliers will be a persistent problem as PKI develops across all these

enterprises. For some time both paper-based and electronic systems will have to coexist.

Implementing PKI in a military MTF environment is expected to substantially increase information integrity, availability, confidentiality, and non-repudiation without substantially increasing the burden on the information management staff. It is likely that the use of public key encryption to secure electronic commerce will increase. If PKI is required in the future for interaction among members of the healthcare community, a PKI implementation that provides interoperability and cross certification capabilities across different enterprises will be necessary and integration of PKI into existing applications will be required.



**Current and Emerging PKI Tools and Techniques****Overview of PKI Tools and Techniques**

Cryptography, and specifically secret key cryptography, is as old as writing itself. Secret key cryptography is also known as symmetric key since the same secret key is used to both encrypt and decrypt. Almost all the examples of cryptographic systems that are familiar are secret key systems. The most widely used of these systems today for unclassified information is the Data Encryption Standard (DES) which is based on the NIST specifications in FIPS PUB 46-2. Because the 56-bit key DES has been successfully attacked using brute-force techniques, it is being replaced by the Triple-DES which still uses the DES algorithm repeatedly with several 56-bit keys and is described in the revised FIPS PUB 46-3.

Public key cryptography (PKC) on the other hand is relatively new having been proposed by Diffie and Hellman in 1976. The first practical implementation, and the one that has stood the test of time, was developed by Rivest, Shamir, and Adleman in 1977 and patented as the RSA algorithm. PKC uses two different keys—a public one for encryption and a private one for decryption—and is therefore an asymmetric key system. The security of the RSA system and PKC in general is based on the difficulty in calculating the secret decryption key from a knowledge of the public encryption key and vice-versa. Such a calculation is not impossible but said to be computationally infeasible given the technology of today and appropriately large key sizes.

**Encryption Algorithms**

The following sections provide some details about current PKC systems and how they are being used today. These sections are not meant to be a mathematical treatise on PKC but rather an overview of the important principles and capabilities.

**RSA Algorithm**

As noted above, the RSA algorithm was developed by Rivest, Shamir, and Adleman and patented in 1977. A company, RSA, Inc., was formed to develop and market products based on this system. One of the first applications was a secure mail product called MailSafe. Other software vendors were reluctant to develop products based on RSA because of the patent and licensing issues. These issues probably led to the development of a NIST standard for digital signatures based on another algorithm. Digital signatures are an additional feature of PKC and will be discussed in a following section.

*Appendix A to Public Key Infrastructure*

The security of the RSA algorithm lies in the computational difficulty in determining the two large prime number factors of a very large composite number, greater than 200 digits. The process is explained briefly as follows.

To generate the key pair first choose two large random prime numbers (greater than 100 digits each),  $p$  and  $q$ . Compute:

$$n = pq \text{ (called the modulus)}$$

Next choose an encryption key  $e$  that is relatively prime to  $(p-1)(q-1)$ . Finally, compute the decryption key  $d$  such that:

$$ed = 1 \text{ [mod}(p-1)(q-1)]$$

The numbers  $n$  and  $e$  are the public elements and the decryption key  $d$  is kept secret. In order to determine  $d$  one must be able to factor  $n$  into its components  $p$  and  $q$ . Recently, RSA-155, a 512 bit number about 155 digits long, was factored in 7 months using about 300 fast SGI and SUN workstations and Pentium PCs and a Cray supercomputer. With computing power doubling every 5 years, this emphasizes the need for longer length keys.

To encrypt using the RSA algorithm, the message (file, etc) is divided into blocks of characters smaller than  $n$ . Each encrypted block will be the same size and calculated by

$$c_i = M_i^e \text{ mod } n$$

To decrypt, each block is calculated as

$$M_i = c_i^d \text{ mod } n$$

The RSA algorithm is much slower than the symmetric DES algorithm in software and even slower in a hardware implementation. For this reason it is used today primarily for key distribution. The message is encrypted with a random key using a symmetric algorithm like the DES or the International Data Encryption Algorithm (IDEA). This key is then encrypted using the public key of the recipient. The recipient receives the encrypted message and the encrypted key. They apply their secret key and extract the message key and use it to recover the message.

The original RSA patent is due to expire in 2000. This may lead to an even wider implementation of the RSA for of PKC. Though not a national standard, RSA is a defacto standard for PKC around the world.

**El Gamal Method**

The El Gamal algorithm is another approach for PKC and though it can be used for encryption, it is primarily important for providing digital signatures. The security of this scheme is due to the difficulty of calculating discrete logarithms in a finite field.

To generate a key pair, one chooses a prime number  $p$  and any two random numbers  $g$  and  $x$  that are both less than  $p$  and calculates

$$y = g^x \text{mod} p$$

The public key is  $y$ ,  $g$ , and  $p$  and the private key is  $x$ . Both  $g$  and  $p$  can be shared among a group of users. Each user would have a unique value for  $x$ .

**Elliptic Curve Cryptography**

Elliptic curves are a type of algebraic function that has been known and studied for over 150 years, primarily for mathematical interest. Only recently have they been applied in algorithms for factoring integers, proving primality, and more importantly, in public key cryptography. Elliptic curves were first used in cryptography in 1985 to implement existing public key algorithms. Since then elliptic curve cryptography (ECC) has been applied to digital signatures as described in ANSI X9.62: The Elliptic Curve Digital Signature Algorithm, and to a number of other cryptographic systems.

While the mathematics for the application of ECC are complex and beyond the scope of this paper, the general defining equation for elliptic curves is given by:

$$y^2 = x^3 + ax + b$$

Where  $a$  and  $b$  are coefficients that are further constrained, certain mathematical operations are also defined, such as addition and multiplication. Whenever you add any two points on a given curve, the result will be either another point on the curve or a special point in space called the "point at infinity."

Mathematics aside, the most important attribute of ECC is the fact the security comparable to other PKC systems can be obtained using keys that are substantially smaller in the case of ECC. According to the IEEE P1363 working group, the effort needed to crack a 254-bit ECC system is about the same as required for a 2800-bit RSA-type system and amounts to about  $5.5 \times 10^{24}$  MIPS years. Furthermore, since ECC systems require additions in their computation, as contrasted with exponentiation for conventional PKC systems, they can be expected to be between four and ten times faster.

Elliptic curve cryptosystems offer the highest strength-per-key-bit of any known public key system. The smaller key sizes result in smaller system parameters, smaller public key certificates, bandwidth savings, faster implementations, lower power requirements, and smaller hardware processors. Low cost implementations of ECC are feasible in restricted computing environments, such as smart cards. Commercial products that use ECC have been developed by Certicom Corporation.

### Digital Signatures

As noted previously, the RSA public encrypting key and the private decrypting key are reciprocal, i.e.

$$ed=1 \bmod [(p-1)(q-1)]$$

What this means in a practical sense is that you can encrypt with either one and decrypt with the other, i.e.

$$D[E(M)] = E[D(M)] = M$$

This property, which is general to all PKC systems, has a useful application. Consider the following scenario. User A has keys  $D_A$  and  $E_A$ . User B has keys  $D_B$  and  $E_B$ . User A creates a message  $M$  to send to user B and encrypts it with user B's public key. User B applies his private key and decrypts it.

User A	Public Keys	User B
Private Key $D_A$	$E_A, E_B$	Private Key $D_B$
$E_B(M)$	$\rightarrow$	$D_B[E_B(M)] = M$

This is just the normal process for providing confidentiality. Consider now what happens if user A encrypts the message a second time using his private key yielding.

$$D_A[E_B(M)]$$

Since the message is purportedly coming from user A, the recipient who knows user A's public key can apply it to the doubly encrypted message and remove one level of encryption

$$E_A(D_A[E_B(M)]) = E_B(M),$$

since  $E_A$  and  $D_A$  are multiplicative inverses.

*Appendix A to Public Key Infrastructure*

Now user B applies his private key  $D_B$  and retrieves the message  $M$  as before. This process is possible only if the message came from user A since only he knows the private key  $D_A$ . This process therefore produces a non-refutable digital signature of the originator, user A.

Since PKC encryption is slow, doubly encrypting would be slower still. Ideally, one would like to encrypt some unique property of the original message. A method for accomplishing this is given in the following section.

**Hashing Functions**

Hashing functions, or more precisely, one-way hash functions are very useful in cryptography and particularly in forming digital signatures. A hash function takes a variable length input data string and produces a fixed length output string. A one-way hash function only works in one direction. That is, given the hashed value, it is hard to generate the original string. Furthermore, it should be difficult to generate two different strings that hash to the same value. The hashing function does not require a secret key and should be known to all. The necessary characteristics of a good hashing function are as follows:

- Maps any size input to a fixed size output,
- No restriction on the type of input file,
- Fast and easy calculation method,
- Requires only one pass through the file,
- Uses no secret parameters,
- Impossible to reconstruct the original files, and
- A single bit change in the input file should alter the resulting hash significantly.

A number of hashing methods have been proposed over the years but the two that have endured are the MD5 algorithm developed by Ron Rivest (the R in RSA) and the Secure Hash Algorithm (SHA) which is a federal standard specified in FIPS PUB 180.

MD5 produces a 128-bit (16 byte) digest of any input file at a processing rate of about 174k bytes per second. While some weaknesses have been suggested, MD5 is generally regarded as robust and has been widely implemented around the world. SHA produces a 160-bit (20 byte) digest of the input file at a processing

*Appendix A to Public Key Infrastructure*

rate of about 75k bytes per second. No successful attack on SHA has been demonstrated.

Combining the hashing operation with the encryption protocol results in the standard implementation of digital signatures. Specifically, to create a digital signature on a document or file one first calculates the hash of the document or file and then encrypts this hash value with their private key. Anyone who wishes to verify the signature can hash the document or file to obtain its hash value. Next they decrypt the encrypted hash value using the originator's public key. If the two hash values are the same, the recipient is assured that the document or file has not been altered and came from the purported originator. If the two values do not agree then either the originator is not correct and/or the file that was received has been altered.

Thus a *digital signature* is an encryption of the hash of a document or file that has been encrypted with the private key of the originator. It therefore provides both data integrity and authentication. The validity of the digital signature depends entirely on the assurance of the binding of the public key to an individual.

Digital signatures are considered legally binding in a number of states and the federal government. The digital signature standard, FIPS PUB 186-1 (1998), now supports digital signatures that use either the Digital Signature Algorithm with the Secure Hash Algorithm or the RSA algorithm with SHA.

**PUBLIC KEY INFRASTRUCTURE**

Public key cryptography can provide strong user authentication, confidentiality, data integrity, and non-repudiation. However, these services depend vitally on the binding of a user's identity to their public key. Furthermore, a practical implementation of PKC with all its advantages requires a formal infrastructure that provides this binding as well as key generation and revocation, and directory services. The following sections describe a model infrastructure that has been proposed and is being implemented in varying detail by a number of vendors and end users. Some entities and functions can be combined as required.

**Administrative Authorities**

This element of the infrastructure provides the administrative functions including development of policies and procedures, end user enrollment, key generation and certification, and key revocation. Within the PKI structure proposed by NIST it is known as the Certificate Issuing and Management System (CIMS) and includes the components of the PKI that are responsible for these functions and includes the Certificate Authority (CA), the Registration Authorities (RA) and other subcomponents.

The administrative structure being developed within the federal government, including the DoD, has a Policy Approving Authority (PAA) and a Policy Certification Authority (PCA). These entities along with the CA comprise the Policy Management Authority (PMA) which is analogous to the NIST CIMS.

**Policy Approving Authority (PAA)**

The PAA is essentially the entity responsible for the overall policy and practices of PKI. It evaluates and approves policies and provides management of the digital certificates. It may delegate oversight of the PKI operation to the next lower level in the PKI hierarchy, the Policy Certification Authority.

**Policy Certification Authority (PCA)**

The PCA is responsible for developing and issuing policy, approving certificate policies, and reviewing certification practices. The PCA is also responsible for coordinating distinguished names for the Certification Authorities (CA), maintaining a master registration list of all CAs and their policies and for providing policy object identifiers for all approved certificate policies. In some implementations this is also known as the Policy Creation Authority.

**Certification Authority (CA)**

The CA is the “workhorse” of the PKI. It is the entity that issues and revokes certificates for a set of users and is ultimately responsible for their authenticity. The CAs act as trusted third parties and vouch for the contents of the certificate they issue. For this reason the CAs must be managed by an entity with a high level of trust in the user community.

The CA is responsible for the following functions:

- Issuing digital certificates to authorized entities in accordance with the certificate policies and certification practices under which they have been approved to operate;
- Maintaining records that document the procedures followed to ensure the integrity of the key management operations;
- Maintaining and making available public key certificate and certificate revocation information for all certificate users;
- Compliance with policies and legal requirements for emergency key recovery to user private keys used for data encryption;

*Appendix A to Public Key Infrastructure*

- Ensuring that all subordinate Registration Authorities (RA)s and End-user Entities (EE)s are aware of and comply with all established rules and procedures;
- Designation and authorization of RAs;
- Providing to the RAs the necessary software, hardware, and training to generate their own public/private key pair and perform their assigned duties;
- Providing the EEs with the necessary software and training to generate their public/private key pair and comply with all policies and procedures; and
- Maintaining the integrity and security of the CA signing key in a manner compliant with the approved certificate policies and practices.

**Registration Authority (RA)**

The RA is designed to be a point of presence for the CA so that users may appear before it to apply for certificates. This is highly advantageous and separates the functions of certificate issuance and signing, which require access to the CAs private key, and should therefore remain offline for protection from the functions related to verification of the identity and registration information which does not require access to the key but rather high availability. The number of RAs required depends on the size and geographic distribution of the EE community. Each RA is responsible for the following:

- Maintaining the security and integrity of their private signing key in accordance with the approved policies and procedures;
- Providing the necessary software and training to the EEs to allow them to generate their public/private key pairs;
- Maintaining adequate records and audits as required by the policies and procedures of the CA; and
- Generating a new RA certificate in the presence of the CA at least every two years or as required by the policies and procedures.

For the MTF community RAs might be located at each healthcare facility and handle EE registration for it.



**End Entities (EE)**

The EE may be any staff member, employee, contractor, vendor, business associate, or proxy for a computer that has a valid operational need for a public key certificate. The EE must provide to the RA the required proof of identity necessary for the certificate. The EE is responsible for protection of their private keys and must notify the CA or RA immediately in the event of any actual or suspected loss or compromise of their private key. EEs will be required to register and renew their certificates in accordance with the policies and practices of the CA.

**Digital Certificates**

Digital certificates are one of the most important components in the PKI. In reality, all other components exist to support, protect, and validate the authenticity and integrity of the digital certificate. It is the end entity's electronic fingerprint that provides absolute and positive authentication of the entity's identity. The digital certificate is simply a collection of information to which the signature of the CA is attached.

The most widely recognized digital certificate format is that one defined by the ISO X.509 standard. The X.509 version 3, which introduced significant changes to the X.509 standard, provides for the use of a number of standard extensions as well as private community-defined ones. Standard extensions are defined for various purposes including key and policy information, subject and issuer attributes, and certification path constraints. Community-defined extensions may be used to implement policies unique to that community. Most commercial products available today support only version 1 or 2; however, the Federal government has about two dozen pilot PKI projects that use X.509 version 3 certificates with application specific extensions.

The information contained in the X.509 version 3 certificate is as follows:

- Version number of the certificate format
- Certificate serial number
- Signature algorithm identifier for issuer's signature
- Certificate authority's X.500 directory name
- Validity period for the certificate
- X.500 directory name of the subject

- Public key algorithm and public key value for the subject
- Issuer's unique identifier
- Subject's unique identifier
- Extensions, critical and non-critical

All of this information is combined with the certification authority's private key to generate a digital signature that is appended to the certificate. Anyone can verify the content of the certificate using this signature and the published public key of the certification authority.

### **Directory Services**

In order to use PKC, one must obtain the public key of the recipient as contained in their digital certificate through some type of directory service. For a PKI, such a service is defined by the ISO X.500 standard. The X.500 directory is analogous to a telephone directory where one uses a person's name to find the desired auxiliary information, the person's phone number.

The X.500 directory entry can contain a host of attributes such as name, organization, job title, e-mail address, and other information. In addition, an X-500 directory entry can be used to represent an entity such as a computer, a printer, company, nation, or a government. The entry can also contain the certificate specifying the entity's public key. Thus an X.500 directory system must be an integral part of any PKI implementation.

### **Certificate Revocation Lists**

Certificate revocation lists (CRL) provide a method for indicating that a certificate is no longer valid or trusted. The X.509 standard also provides the framework for CRLs. Versions 1 and 2 of the X.509 standard provided for simple CRLs that did not address size issues nor provide a sufficiently granular time stamp capability. As with the Certificates, X.509 version 3 provided for extensible formats for the CRLs. Currently, however, version 2 CRLs are the most commonly used.

The X.509 version 2 CRL contains the following information:

- CRL version number,
- Issuer's signature algorithm identification,
- Issuer's X.500 name,
- Date and time of the list update,

- Date and time of the next list update,
- Listing of all revoked certificates,

Certificate serial number	Revocation date
.	.
.	.
.	.
Certificate serial number	Revocation date

- Issuer's signature formed from a hash of all of the above information and encrypted with the issuer's private key.

**Overview of Current PKI Implementations in Government and Private Industry**

The deployment of a PKI that supports internal applications and provides compatible interfaces to external entities and applications can be difficult to accomplish. There are several approaches, however, that can be used to do this. Each approach has both advantages and disadvantages that must be considered. Basically, an enterprise can build their own PKI or pay someone else to do it for them. The latter case has two additional variations that can be employed.

An enterprise wishing to implement their own PKI with its CA, RAs, and surrounding systems and technologies assumes total responsibility including the staffing requirements and all liability. This approach is the most resource-intensive but does provide the most control over the system. A large institution that only needs to communicate internally or to its own external components might use this approach.

In contrast, an organization can choose to pay a third party to run its CA. In this case, the third party issues certificates on behalf of the organization. This approach has the advantage of using the trained support staff of the provider, its highly secure facilities, and the necessary supporting infrastructure. This approach requires a great deal of trust in the third party as well as in its policies and procedures and therefore gives the organization the least amount of control. A small organization that did not have sufficient internal resources might use this option.

A variation of the previous approach that combines features of the two involves deploying an integrated PKI service. In this case, all the software, hardware, and other components are maintained on the organization's premises while the certificate processing services are provided by a third party. This results in shared investment and shared risk and leverages the expertise and security of the certificate provider. The organization maintains control over policies but leaves the implementation to the provider. In some cases this approach offers the most advantages.

As noted, the best approach for deployment of a PKI can be different for different organizations and environments. As a result there are a number of different implementations being undertaken by private industry and the federal government. The following sections will outline some of these approaches to PKI deployment.

**Private Industry Implementations**

Private industries, unlike the Federal government, do not have the funds necessary for investing in a large scale R&D project to develop a PKI to support their needs. As noted in the preceding section, there are two primary approaches that can be taken to achieve the needed infrastructure. An enterprise can purchase the necessary software and deploy their own CA. It should be noted that while a CA is a major component, it is not all that is necessary to complete the PKI. Nevertheless, using something like the Netscape Certificate Server software, an enterprise can establish a secure web capability within the enterprise. Additional functionality can be achieved by choosing some application such as a secure e-mail that supports X.509 certificates.

The second alternative for an enterprise wishing to establish a PKI is to purchase a complete PKI solution from a vendor such as Entrust Technologies, Celo Communications, or a PKI integrator. Appendix B of this paper gives a listing of a number of the PKI vendors that provide a range of products and services.

Regardless of the approach taken, the main impediments to wide scale use of PKI in businesses are the problems of interoperability and user acceptance. Recognizing these problems, five of the leading vendors (Baltimore Technologies, Entrust Technologies, IBM, Microsoft, and RSA Security) have established a cross-industry PKI Forum. The PKI Forum is an international, not-for-profit, multi-vendor alliance whose purpose is to accelerate the adoption and use of PKI and PKI-based products and services. The PKI Forum advocates industry cooperation and market awareness to enable organizations to understand and exploit the value of PKI.

The PKI Forum will provide an opportunity for vendors to demonstrate support for standards-based, interoperable PKI by developing PKI interoperability profiles based on business-driven requirements in certificate interoperability, directory-PKI interoperability, application interoperability, certificate validation and other areas. The forum will sponsor product interoperability demonstrations and work closely with standards organizations and external test bodies. The forum also will bring customers and vendors together to increase customer knowledge about the value of PKI and demonstrate how PKI solves key security issues for electronic business.

**Federal Government Initiatives**

The complexity of developing and deploying a practical PKI is somewhat evidenced in the variety of organizations, programs, and pilot implementations within the Federal government. Within the different elements

of the government, such as DoD, there are additional variations in the approaches to establish a PKI. Part of the problem, as noted previously, is the fact the PKI standards are still evolving and the various PKI vendors are developing products that do not necessarily interoperate well. This has forced federal agencies to develop their own application-specific solutions. Currently there are about two dozen separate programs in various stages of implementation within the Federal government. The following sections summarize a number of these programs that may impact on the approach that is appropriate for DoD Medical Treatment Facilities.

### **Federal PKI Steering Committee**

The Federal PKI Steering Committee of the Government Information Technology Services Board has the task of providing government-wide guidance and coordination of Federal activities necessary to implement a PKI. Membership of the Federal PKI Steering Committee is composed of representatives from twenty government agencies, departments, and offices. The FPKI Steering Committee is responsible for coordinating, overseeing, monitoring, implementing, and reporting on the development of a PKI that will support secure electronic commerce and electronic mail as well as other Federal agency programs that require the use of PKC.

### **NIST PKI Program**

The National Institute of Standards and Technology (NIST) has taken a leadership role in the development of a Federal PKI that will support digital signature applications and other public key security services. In addition to coordinating with various industry and technical groups such as the Federal PKI Steering Committee and its Technical Working Group, NIST has developed a set of security requirements for validation of PKI components.

NIST also has several laboratory activities that involve industry participants. The Minimum Interoperability Specification for PKI Components (MISPC) project is attempting to define and test various components for their features and to identify a minimum set of desired features that will promote and ensure interoperability. There is also a testbed for this evaluation.

Additional activities include the development of a Reference Implementation and initial implementation of a root CA for the Federal PKI. The purpose of the Reference Implementation is to have a proof of concept for the MISPC that will be available for testing of commercial products. The root CA will be used to examine hierarchical and non-hierarchical CA relationships, scalability, and other operational issues. NIST further plans to create a bridge CA that can be used to certify other government CAs in order to promote interoperability between agencies.

**GSA ACES Program**

The General Services Administration Access Certificates for Electronic Services (ACES) program is designed to provide the technology necessary to secure government business over the Internet by providing digital certificates for authentication. The intent of this program is to provide a government-wide PKI that could be used by the public wishing to conduct business with the government and for the smaller agencies of government that could not afford the time and money to put together their own PKI the way the Department of Defense and other large agencies have done.

Vendors were initially reluctant to bid on providing certificates for ACES because of liability issues until GSA lawyers issued a legal finding that said the government, not ACES vendors, would be liable for any accidental release or misuse of personal data. There are currently three vendors (Digital Signature Trust Co., Operational Research Consultants Inc. and AT&T Corp.) that have been awarded contracts to provide certificates for the ACES program. Prices for certificates drop as volume increases and proposed prices range from 40 cents to \$1.20 per certificate under ACES. The first certificates are due to be issued around the first of CY 2000.

A number of government agencies have expressed interest in using the ACES contract. Among these are the Social Security Administration, the National Institute of Health, and the Education Department. As ACES matures and demonstrates its usefulness, it is expected that there will be more participants. One additional driving force is likely to be the Government Paperwork Elimination Act (GPEA). The GPEA requires all agencies to provide the public with the option of submitting government forms electronically whenever possible by October 2003. Congress has passed a bill that provides a legal basis for documents using electronic signatures and is working on several others. This will further stimulate interest among government agencies in using ACES.

**DoD Initiatives**

The Defense Information Systems Agency (DISA) has taken the lead in developing PKI policies for the Department of Defense (DoD). Recognizing that PKI technology is still immature in the marketplace and changing rapidly as the standards evolve, DoD has adopted a strategy for early implementation of PKI technologies and active participation with industry to define technical needs and issues, resolve interoperability problems and develop a government-wide PKI that will support all services and agencies.

The DoD PKI is being developed to provide certification services based on the following policies and requirements:

- Standards-based;
- Support multiple applications and products;
- Provide secure interoperability throughout DoD, other Federal government agencies and with industry;
- Support digital signature and key exchange/encipherment;
- Provide functional separation of keys;
- Support key recovery/data recovery;
- Support legal non-repudiation;
- Commercial-based, allowing for the possible outsourcing of elements in the future; and
- Support FIPS-compliance requirements.

One of the first implementations of the DoD PKI will be in support of the Defense Travel System (DTS). This pilot program is based on X.509 version 3 certificates and version 2 Certificate Revocation Lists and can be used by DoD employees for secure transactions of the DTS.

DISA has initially selected two vendors to supply and manage the digital certificates for the DoD under the Interim External Certificate Authorities (IECA) contract. Operational Research Consultants, Inc., and Digital Signature Trust Co. will provide certificate management services for the DTS for up to a year after which DISA will put in place permanent external certificate authorities.

In addition to the DTS application, DoD plans to use the IECAs to secure communications for the Electronic Document access system, which will enable vendors to access DoD solicitations and modifications over the Internet; and the Paperless Contracting Wide-Area Work Flow, which will collect documents and forms from vendors and DoD sources. The number of users for these three applications could grow to over 300,000 in year 2000 according to DISA.

The Defense Logistics Agency is planning to use PKI in support of their Standard Procurement System (SPS) deployment. The SPS will provide a single DoD procurement system with electronic commerce capability. Using a COTS database application the SPS will process sensitive but unclassified information and has the goal of reducing operating expenses of the DoD procurement infrastructure.



*Appendix B to Public Key Infrastructure*

One of the largest and most technically mature programs in the DoD is the Multilevel Information Systems Security Initiative (MISSI) of the National Security Agency. To support DoD customers and gain technology expertise, the PKI functions have been implemented through a government-sponsored CA Workstation (CAW) that manages keys, privileges, and certificates. The current operational MISSI CAW hierarchy focuses on supporting FORTEZZA Crypto Cards. In the near term, the Defense Information Systems Agency (DISA) and NSA as part of MISSI are utilizing the CAW and FORTEZZA Crypto cards to field a high assurance PKI for the Defense Messaging System (DMS) for organizational messaging traffic. This implementation supports the required DMS user security services through the use of the FORTEZZA Crypto card and an X.500 directory. A long-term goal of MISSI is to support customer PKI needs, through a combination of (1) acting as an unbiased authority validating the security goodness of the commercial PKI products/services, and (2) driving the development of robust PKI solutions by specifying requirements and transferring technology expertise to industry.

Additional information about FORTEZZA and smart cards and how they can be used to support PKI implementation is given in Appendix C.

**Other Government Projects**

**DOE.** The Department of Energy has three different projects underway that will utilize PKC for digital signatures and privacy. The Electronic Research Administration Demonstration project will test emerging security technologies for electronic data interchange using secure E-mail based on Internet standards. This project, which involves six Federal agencies and eight academic research organizations, will test the interoperability of multiple vendors' products across an open system environment. The initial implementation focuses on providing encrypted electronic grant applications and providing key recovery services.

At the DOE Lawrence Livermore National Laboratory, the Badges and Security Clearances project uses the PKI digital signature capability to sign and route, via E-mail, an electronic form used to request changes in clearance and badge status for an individual. The form is signed by the originating department, Human Resources, and the form and signature are verified by the Security Department.

Also at LLNL the Public Key Infrastructure program is building the PKI to support both business and programmatic operations. Part of these operations include network interactions with many other DOE laboratories, facilities, and vendors. In order to utilize the network, LLNL's infrastructure must provide strong authentication, non-repudiation, message integrity and privacy for the

information being exchanged. An emergency access capability is viewed as a critical part of this infrastructure if the full potential of the public key encryption technology for privacy is to be realized. LLNL will exercise the key recovery capabilities of a commercial software product to determine its ability to meet requirements.

**Treasury.** The Department of the Treasury Secure Electronic Messaging System is a pilot project aimed at exploring the use of PKI tools to facilitate the flow of procurement information. Specifically, the project is designed to reduce the time and resources needed to request, review and approve task orders issued under existing contracts. In the case of this pilot, OTM has instituted a system that enables Treasury employees to interact securely with key employees of a Treasury contractor over the Internet. Encryption, digital signatures, and key recovery functionality are being used to secure all transactions.

A second Treasury program supports the requirements of the North American Free Trade Agreement (NAFTA). Designed to streamline clearance of commercial goods through the customs process, the prototype electronic commerce initiative is intended to demonstrate how the NAFTA customs and trade processes could function in a cost effective, secure manner. Designated as the North American Trade Automation Prototype (NATAP), the initiative includes live operations at US, Mexican, and Canadian frontiers. While the prototype is low volume relative to the daily cross-border activities, it has been implemented by over 400 trade partners in the North America, and therefore represents a unique exploration of state of the art technology by government and commercial organizations.

Since the application has been implemented using the Internet, it has been designed with a high level of security features to ensure confidentiality and authenticity of business sensitive information contained in routine customs declarations, while preserving the availability of that information through key recovery.

**DOT.** The Department of Transportation along with thirteen other Federal agencies have joined in development of a comprehensive Electronic Grants System (EGS). This system is designed to streamline the federal grant process. EGS uses Java and Information Broker technologies to enable any grant customer to electronically exchange grant data with any federal agency, or multiple agencies, using a single WWW user interface. These technologies also provide a truly interactive user interface by transmitting grant data to and from federal databases in near real-time.

With funding from the GITS Board's Key Recovery Demonstration Project, secure features including digital signatures, encryption and key recovery, have

*Appendix B to Public Key Infrastructure*

been added to the EGS pilot. This secure pilot is currently being tested with state government and university partners. Following completion of this testing, full EGS development will begin.

**FBI.** The Federal Bureau of Investigation (FBI) Internet Communications Security project's goal is to develop a method for the Computer Investigations and Infrastructure Threat Assessment Center to communicate with representatives of private industry, the academic community, and other law enforcement agencies in a secure manner, while at the same time supporting key recovery. The pilot will test secure E-mail over the Internet and will involve security services such as digital signature, encryption, and CA.

A second FBI program that will require PKI support is the Secure E-mail project. This project seeks to secure the contents of E-mail messages sent within the FBI and between the FBI and other Department of Justice components. This project focuses on a small number of E-mail users to demonstrate the use of encryption for confidentiality and feasibility of providing key recovery for e-mail messages.

**Agriculture.** The Department of Agriculture has several programs that will use PKC technology. The Electronic Benefits Transfer initiative will employ security technologies in the Food and Consumer Services security demonstration system. After a successful demonstration, the FCS will use the technologies in a live benefits environment. This will be done by selecting states in which the security technology will be integrated into the operational EBT system.

A second program within the Department of Agriculture that will utilize PKI technology is the National Finance Center (NFC). The National Finance Center project focuses on the requirements and functions necessary to allow the NFC to support digital signature requirements from client agencies and to have the NFC's CA participate in interoperability testing with NIST as a trusted CA site using COTS software. Selected application documents, that are electronically submitted and require an original signature, are transmitted to NFC with an attached digital signature. The digital signature is stored with pertinent document information prior to application processing. Along with this initial pilot, NFC is establishing IPsec Encryption with VPN technology using certificates with various applications at NFC. NFC has begun interoperability testing with NIST. Further interoperability testing with NIST and other designated agencies will be conducted. Hardware, software, and cryptographic modules upgrades and enhancements will be implemented to ensure that NFC's goal of offering a CA that complies with FIPS 140-1 Level III requirements is realized.

**DOC.** In addition to the NIST PKI program, discussed earlier in section 4.2.2, the Department of Commerce has several more PKI related programs at NIST and also at the National Technical Information Service and the Patent and Trademark Office.

The Information Technology Laboratory at NIST has been focusing on the design, implementation and use of advanced systems for cryptographic based computer security and office automation systems. The Purchase Order Request System combines both technologies into a system which will provide NIST with basic infrastructure components necessary for migration into a paperless process. The system allows users to generate and digitally sign purchase order requests, which are then routed to an approving authority. The approving authority reviews the request, verifies the user's signature and, if in agreement with the request, signs it and sends the electronic form to the administrative officer for processing. The system has been developed in a modular fashion so that both the digital signature module and the certificate management module can be used to support other applications using digital signature technology.

The NIST Root Certification Authority Reference Implementation project's focus is development of an initial implementation of a top level or root CA for the PKI and to conduct experiments with Federal agencies who are actively engaged in the development and use of digital signature technology. The CA will issue certificates to various pilot CA applications now being developed by Federal agencies. This test will foster a flexible hierarchy, which could support agency-level digital signature based applications. The initial root CA will also be used to cross-certify with CAs operated or provided by commercial vendors.

The National Technical Information Service (NTIS) FedWorld Secure Web and CA Project has prototyped trusted-agent services that support digital signature, encryption of files and messaging, and authorized emergency access to encrypted information through key recovery management. NTIS currently provides the security infrastructure supporting the DOT Electronic Grants System and the NIH Electronic Grants Pilot and is actively exploring partnerships with other agencies that need digital signature and encryption capabilities incorporated into current or planned web-based applications.

**Patent and Treasury Office.** The Patent and Trademark Office (USPTO) of the Department of Commerce has two programs in progress that will use PKI technology. The USPTO International Patent Document Exchange Project will demonstrate the exchange of patent documents in secure electronic form between the Trilateral Offices (USPTO, European Patent Office, and Japanese Patent Office) and the International Bureau of the World Intellectual Property

*Appendix B to Public Key Infrastructure*

Office (WIPO) to reduce processing costs and the burden of applicants. The pilot implements encrypted e-mail with a key recovery capability.

The second USPTO project, the Electronic Patent Application Filing System (EPAFS), was started as a project designed to help determine the feasibility of using the Internet and the WWW as a platform for electronic filing of patent applications. The development of EPAFS was initially focused on the filing of applications under the Patent Cooperation Treaty, a format that has international applicability. Currently, extensions and enhancements are being made to EPAFS for demonstration in conjunction with the participation of the USPTO and the Key Recovery Demonstration Project under sponsorship of the GITS Board. Recent developments in the areas of high-grade encryption and digital signature technology coupled with emerging vendor products and software components offer promising solutions to many of the problems associated with moving to an electronic patent filing and processing environment. For example, encryption key recovery interchange standards and software modules will not support and enhance the recovery capabilities of archival data and other information assets. General business practices will be enhanced with this new key recovery capability. The USPTO will then be able to concentrate efforts on delivering better services to the Intellectual Property practitioners and general IP community now that certain security requirements such as data archival/retrieval, strong encryption, and non-repudiation can be met with commercially available key recovery and digital signature products and service offerings.

**GSA.** The General Services Administration (GSA) Paperless Federal Transactions for the Public pilot was an early attempt to develop a user friendly, interoperable solution utilizing Federal and industry standards, where available, to provide a public key architecture that would satisfy the concerns of the regulatory agencies for performing electronic commerce and funds transfer over the worldwide web. By developing partnerships with industry and concentrating on COTS products, a proof-of-concept pilot was designed and implemented using public key technology to enable digital signature and encryption. The concept called for an architecture requiring industry partners to adhere to a set of Federal standards, including the use of the DES encryption algorithm, and DSS for digital signatures. Six agencies agreed to participate and develop applications for use with the Paperless pilot. Each was provided a web server, server software, client software, and hardware tokens containing their key pairs. The pilot completed its first successful implementation when the GSA FTS2001 procurement was conducted in a completely paperless environment.

**NASA.** The National Aeronautics and Space Administration is also involved in PKI development through its Public Key Infrastructure project. The PKI

*Appendix B to Public Key Infrastructure*

pilot project will investigate the use of standard X.509 certificates for public key infrastructure, the deployment issues related to the PKI, and other key infrastructure, such as Simple Public Key Infrastructure/Simple Distributed Security Infrastructure (SPKI/SDSI). The study of the PKI includes the COTS implementation of the PKI technology. The deployment issues related to the PKI (i.e., scalability, reliability, performance, maintainability, and cost) will be investigated. Additionally, the Ames Research Center (ARC) will study the latest trend in the area of the PKI, such as SPKI/SDSI. The study of SPKI/SDSI covers the research cooperation between ARC and Professor Ronald Rivest of MIT. SPKI is led by Carl Ellison. SDSI is designed by Ronald Rivest and Butler Lampson. SPKI/SDSI emphasizes unique approaches to naming and delegation of authority. It will be useful for authorization and access-control applications. A SPKI/SDSI version 2.0 prototype is being implemented at the Massachusetts Institute of Technology.

**NIH.** The Electronic Research Administration of the National Institute of Health (NIH), recognizing that all secure systems require that trust be established between users, has developed the ERA Public Key Infrastructure pilot program. The NIH pilot enables grantee organizations to exchange data with NIH in a trusted manner. Many grantee organizations participating in the pilot with NIH have implemented grant administration systems. These systems are capable of generating grant administration data in a standard encoded format. The NIH pilot provides these organizations with a non-proprietary means to submit the encoded data securely to NIH. More specifically, the NIH pilot demonstrates how World-Wide Web Electronic Data Interchange and PKI technologies can be integrated to create a secure electronic grants system. For PKI services NIH has partnered with the National Technical Information Service (NTIS). NTIS received funding from the Government Information Technology Services Board Key Recovery Demonstration Project to prototype Certification Authority and Key Recovery Agent functions for the NIH pilot. As part of its PKI service, NTIS has supplied a workgroup security tool to NIH pilot participants. Using this GUI-based tool, grantees can digitally sign files. These files can then be uploaded to NIH via a Secure Socket Layer (SSL) channel. Additionally, grantees can use the tool to encrypt data on their local systems. The NIH pilot is part of an overall NIH effort to work with other Federal agencies and the research community to develop an electronic research administration system that promotes standardization of data and flexibility in non-proprietary technological solutions.

**SSA.** In order to streamline the process of wage reporting by small businesses, the Social Security Administration entered into a proof-of-concept (POC) demonstration project with Pitney Bowes, Inc. to test the usefulness of the Internet as a means for small businesses to submit their W-2 and W-3 data

---

*Appendix B to Public Key Infrastructure*

to SSA. The annual wage reporting pilot demonstrated that trusted third party such as Pitney Bowes could work closely with SSA on an ongoing workload without compromising the privacy or security of the information that SSA is charged with processing, and that data can be moved over open networks while employing the encryption and other data security technologies. The pilot was an important opportunity to learn more about small business customers, third parties providing secure authentication and certification services, the PKI, key recovery services, and other Internet issues.



**Smart Token Technologies**

While they are not considered components of the PKI, there are a number of smart token technologies that can be implemented in the PKI that will provide valuable support and operational efficiency. Normally ones digital certificate with its public key and their private key are stored on their computer. This approach does not work well when users are mobile. Putting this information on a floppy disk is risky even if some form of protection such as encryption is used. Smart tokens provide a way to safely store ones public key credentials and can also be used to perform the encryption needed for digital signatures and privacy.

**FORTEZZA PCMCIA Card**

The DoD Defense Message System is built around the FORTEZZA Crypto Card which is in a PCMCIA Type II format and provides authentication and encryption services. The DoD has already issued several hundred thousand of these cards for the DMS and the Navy is planning to buy up to 150,000 more over the next five years to support the Space and Naval Warfare Systems Command System Center's public key deployment. The FORTEZZA card requires a PCMCIA reader slot in the PC and many computer systems now provide this. An external reader is also available. Prices for the card and the reader vary considerably but are generally in the neighborhood of several hundred dollars each. The advantage in using the FORTEZZA card is that a number of applications have been developed to use it including secure electronic mail and a secure web browser.

The FORTEZZA card stores the user's keys and the corresponding certificate information. The crypto engine in the card is suitable for encrypting sensitive, unclassified information. A version of the card with a different algorithm can be used to encrypt secret level information. Activation of either type of card requires the use of a personal identification number (PIN). After a number of incorrect PINs, the FORTEZZA card will automatically deactivate and requires the CA to unlock it. These measures prevent unauthorized use of the card, should it be lost or stolen.

**Smart Cards**

Smart cards are credit card size tokens that have a microprocessor embedded in the card as well as some data storage capacity. These cards are being studied by the General Service Administration (GSA) for integration with PKI technology to provide authentication, digital signatures, and access control for certain applications on a government-wide basis.



***Appendix C to Public Key Infrastructure***

The GSA will look at whether the emerging Federal Bridge Certification Authority concept can provide a common point of trust for agencies using smart cards and PKI technology for access controls. The bridge CA will be a centralized entity designed to distribute digital signatures between agencies to ensure that their respective CAs and certificates will interoperate.

By the first of CY 2000 the GSA will release a solicitation for its government-wide Smart Access Common ID card procurement, which vendors have valued at more than \$1 billion.

In addition, in November 1999 Deputy Secretary of Defense John Hamre released a memorandum that directed DoD components to use a smart card – based common access card that will be the standard identification card for military, civilian, and contractor employees. These cards will be used to gain access to buildings and to DoD computer networks and systems.

While smart cards currently do not have the computational capability of PCMCIA cards like FORTEZZA, they are much cheaper and do have enough memory to store public key certificates and other ID information and access credentials. Smart cards also require a special reader but these are also relatively inexpensive. Cost of an average smart card is under \$20 each. Some cards have been produced using elliptic-curve cryptography that cost about \$4 each.

Smart cards with 16K and 32K of memory as well as a 16-bit microprocessor are expected to be available in CY 2000 and 32-bit chips are under development. The higher speed and enhanced memory will enable smart cards to execute high-level languages and perform computationally intensive tasks.

Because of their size and format, smart cards can easily be incorporated into personnel photo ID badges. These badges could also hold public key credentials and access control data in a single token.

**Vendors and Products**

All the advantages of PKC for electronic commerce can only be realized if it is supported by an appropriate PKI. Vendors for PKI products and technology have realized this as evidenced by their rapid growth over the last several years from about three to nearly three dozen different companies. The revenue for PKI products and services is expected to be around 1.1 billion dollars by the year 2003. Two years ago, in 1998, the total revenue was around 100 million dollars.

The following is a listing of current PKI vendors and the services and products they provide, taken from the June 1999 issue of *Information Security* magazine, the August 1999 issue of *SC Info Security News Magazine* and other sources.

**Baltimore Inc**  
**[www.baltimore.com](http://www.baltimore.com)**

Provides CAs that enable companies to build a PKI, as well toolkits to PKI-enable apps.

**BCE Emergis**  
**[www.bceemergis.com](http://www.bceemergis.com)**

OnWatch Service utilizes the Entrust/PKI to store, distribute and manage encryption and authentication keys via the 'Net.

**Bull Worldwide Information Systems**  
**[www.us.bull.com](http://www.us.bull.com)**

Compatible with other vendors' CAs, PKIs perform certificate registration and CA functions.

**Celo Communications Inc.**  
**[www.celocom.com](http://www.celocom.com)**

Solutions combine user authentication and digital signing over SSL, PKI-enable legacy and client/server intranet applications and add digital signature capability to browsers.

**CertCo**  
**[www.certco.com](http://www.certco.com)**

Provides secure root CA services.

**Choreo Systems Inc.**  
**[www.choreosystems.com](http://www.choreosystems.com)**

Provides PKI consulting and technical and implementation services.

**Chrysalis-ITS****[www.chrysalis-its.com](http://www.chrysalis-its.com)**

Solutions to offload cryptographic processing from CAs in a PKI.

**CyberSafe Corp.****[www.cybersafe.com](http://www.cybersafe.com)**

Provides outsourced CA and issues certificates for authentication and access control.

**CygnaCom Solutions Inc.****[www.cygnacom.com](http://www.cygnacom.com)**

Offers integration and consulting services with PKI products from Baltimore, Entrust, GTE CyberTrust and VeriSign.

**Cylink Corp.****[www.cylink.com](http://www.cylink.com)**

Builds PKIs and provides security components for PKIs.

**Digital Signature Trust Co.****[www.digsigtrust.com](http://www.digsigtrust.com)**

PKI solutions include digital certificate life-cycle management, CA services, repository and business process reengineering services.

**Diversinet Corp.****[www.diversinet.com](http://www.diversinet.com)**

Passport Certificate Server offers full life-cycle functionality for digital certificate management, providing PKI-based security for wireless, smart card and cable environments.

**E-Lock Technologies Inc.****[www.elock.com](http://www.elock.com)**

Offers free PKI security software that can issue an unlimited number of digital certificates with unlimited life.

**Entegrity Solutions****[www.entegrity.com](http://www.entegrity.com)**

Provides PKI application integration. Partner with Cybertrust, Verisign and ValiCert.

**Entrust Technologies Inc.****[www.entrust.com](http://www.entrust.com)**

Entrust/PKI, now in its fifth version, is in-house-managed PKI software for secure electronic transactions, desktop and developer applications, access control devices and Internet commerce.

**GTE CyberTrust****[www.cybertrust.com](http://www.cybertrust.com)**

Provides outsourced PKI products and services that allow organizations to issue and manage digital certificates for network-based applications.

**IBM Corp.****[www.ibm.com](http://www.ibm.com)**

Provides CA and key services, as well as PKI reference code implementation, which can be used to build in-house PKIs.

**iD2 Technologies****[www.id2tech.com](http://www.id2tech.com)**

Solutions combine PKI and smart cards.

**IFsec****[www.ifsec.com](http://www.ifsec.com)**

Consulting and integration services for PKI.

**KyberPASS Corp.****[www.kyberpass.com](http://www.kyberpass.com)**

Products PKI-enable applications and allow an organization to implement PKI technology to operate their own CA.

**Labcal Technologies****[www.labcal.com](http://www.labcal.com)**

Offers a CD-ROM package that acts as a guide to implementing a PKI in an organization.

**Litronic Inc.****[www.litronic.com](http://www.litronic.com)**

Provides 32-bit PKI smart cards, and solutions that combine PKI and smart card technology into enterprise-wide solutions.

**LockStar Inc.****[www.lockstar.com](http://www.lockstar.com)**

PKI-to-legacy integration technology provides digital certificate-based user authentication and secure data transfer between browsers and back-end legacy systems.

**Lucent****[www.lucent.com](http://www.lucent.com)**

Integration of VPN technology and IPSec clients with PKI certificate solutions from Entrust and Verisign.

**Microsoft Corp****[www.microsoft.com](http://www.microsoft.com)**

Provides services for creating, deploying and managing PKI-based applications.

**Network Associates Inc.****[www.nai.com](http://www.nai.com)**

Net Tools PKI is a solution for issuing and managing X.509 certificates for Network Associates' products.

**nCipher Inc.****[www.ncipher.com](http://www.ncipher.com)**

Offers nFast cryptographic accelerators.

**Phaos Technology Corp.****[www.phaos.com](http://www.phaos.com)**

The Centuris PKI Platform is a Java-based solution for building PKI applications.

**Rainbow Technologies Inc.****[www.rainbow.com](http://www.rainbow.com)**

Provides PKI protocol engines and a USB-based cryptographic personal authentication token for Web-based access control, securing e-mail and VPNs.

**Security Dynamics Technologies Inc.****[www.securitydynamics.com](http://www.securitydynamics.com)**

Keon, a family of PKI products based on RSA's public-key technology, provides PKI solutions and enables customers to build their own PKI solutions.

**Shiva****[www.shiva.com](http://www.shiva.com)**

VPN implementation with certificate authority and authentication services.

**SHYM Technology Inc.****[www.shym.com](http://www.shym.com)**

PKEnable integrates packaged or legacy applications with multiple PKI services.

**Sirrus Internet Solutions****[www.sirrus.com](http://www.sirrus.com)**

Provides policy management for authentication and authorization through integration with existing PKIs.

**SPYRUS****[www.spyrus.com](http://www.spyrus.com)**

Provides infrastructure components including PAA, PCA, CA, and RA. Public key smart cards and FORTEZZA PCMCIA cards and readers are also available.

**Thawte Certification****[www.thawte.com](http://www.thawte.com)**

Offers software and solutions that help organizations create and deploy PKI solutions.

**ValiCert Inc.****[www.valicert.com](http://www.valicert.com)**

Validation authority (VA) products and services ascertain digital certificate trust by adding validation to PKI applications.

**VeriSign Inc.****[www.verisign.com](http://www.verisign.com)**

Offers outsourced PKI software and processing services through its OnSite PKI platform, now in its fourth version.

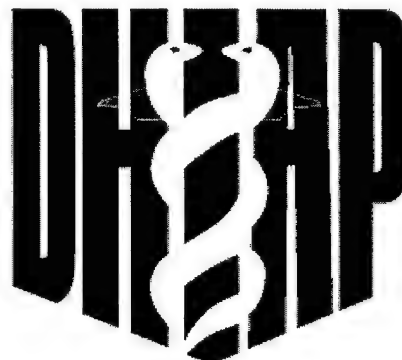
**Xcert Int'l. Inc.****[www.xcert.com](http://www.xcert.com)**

Offers solutions that PKI-enable Web servers and integrate products with a CA and PKI.

# Trust Model

---

## Defining and Applying Generic Trust Relationships in a Networked Computing Environment



Authors: Dr. Jack A. Stinson, Jr., ATI  
Stephen V. Pellissier, ATI  
Archie D. Andrews, ATI

May 2000

ATI IPT Special Report 00-07

This work is supported by the U.S. Army Medical Research and Materiel Command under Contract No. DAMD 17-99-C-9001. The views, opinions, and/or findings contained in this report are those of the authors and should not be construed as an official Department of the Army position, policy, or decision unless so designated by other documentation.

## 1. INTRODUCTION

The Internet was created in an open and trusting environment, designed to facilitate the free exchange of technical collaborative information. While this open and trusting model may no longer apply, framing necessary relationships in terms of trust and degrees of trust can assist in understanding and managing the risks that are inherent in modern computer networks and systems. This paper examines the concept of a "trust model" in an integrated computer environment (local and external networks and computers) and its associated impact on risks.

A trust model is a tool that helps one visualize and understand the degree of confidence that is intentionally or unintentionally granted to individuals, computer networks, and systems, based on the associated risks that are inherent with granting this confidence. The more completely the trust model is defined, the greater awareness one will gain of the threats and vulnerabilities and especially the risks based on those threats and vulnerabilities. In light of the trust model and knowing these risks exist, one can then assess the risk and decide to implement a solution to mitigate the risk, or depending on cost, resource availability, or technology, one may choose or be forced to accept the risk.

The concept of a trust model is the alternative to the "deny model" that is currently widely used in the computer field. A deny model is a description of those individuals, computer networks, and systems that are refused or not granted access. While this model can theoretically be as correct as the trust model, it is inherently incomplete because one must specifically identify each entity that is denied access. With millions of computers on the Internet and hundreds of new sites brought online each day, it would be nearly impossible to keep the deny model up to date, thereby causing gaps in the protection it could provide and increasing risks. Since the trust model describes who is let-in rather than who is kept out, it is more likely to remain current while exposing unintentional gaps and thereby enhancing risk mitigation.

This paper begins in Section 2 with illustrations of two commonly encountered systems and relates those to a model of the trust implicit in the operation of these systems. Specifically, we examine a publicly available web-server, the trusts implicit in such a publicly available entry point to an organization, and the impact when that trust is violated. As a second illustration, we analyze the trusts associated with a remotely administered system. Remote administration of systems and applications from centralized sites allows economies of scale and efficient use of specialized skills. It is becoming more prevalent as a de facto method of operation. Again, we will examine the trusts implicit in such an arrangement and the impact when that trust is violated. These illustrations are meant to motivate the reader to appreciate the concept of the trust model as a tool for understanding and managing risks inherent in highly connected computer systems. Section 3 provides the fundamentals of a trust model, examining trust at various levels of computing system configurations, beginning with the trust inherent in a modern computer system, and building to the



trust models imposed on highly interconnected and interdependent systems. Section 4 builds on the application of trust models by revisiting the illustrations of Section 2 in view of the fundamentals laid out in Section 3. Potential modifications to the illustrated system configurations are examined, along with an analysis of how to apply the trust model concepts to system and risk management. Appendix A contains the essential details used in the development of the web-server trust model provided in Section 4.

## 2. TRUST IN COMMONLY ENCOUNTERED SYSTEMS

The illustrations in this section examine two common systems in an integrated computer environment, a web-server and a remotely administered host, and provide a framework for the developing applicable trust models. The illustrations start with an overview of typical system usage and the explicit and implicit trust models inherent with this usage. Next, the risks associated with these trusts are described and possible exploits are explored. Finally, changes to the trust model that will help mitigate these risks are suggested. By understanding the trusts that are explicitly or implicitly invoked, the risks associated with these trusts can be determined and managed to reduce possible system exploitation.

### 2.1 Web-Server Illustration

A web-server that provides information about a hospital, or serves the hospital community, must be publicly available for access over the Internet. It may display periodic updates on current hospital news, available services, normal hours of operation, points of contact, and phone numbers. It may also provide access to forms that patients can complete and submit for the purpose of either gathering information related to their appointments and insurance or for voicing comments and concerns or even for ordering prescription refills. The web-server could also provide links to other healthcare providers, healthcare information, and regulatory agencies. It could have a section on frequently asked questions (FAQs) about diseases and downloads of information or healthcare related computer programs. It may even have a restricted area that only hospital personnel can access. Such a restricted area may include employee phone lists, meeting notices, notes to the staff, and job listings. By determining the state of the web-server in terms of the trusts that are invoked, the associated risks can be identified and mitigated. The explicit and implied trusts invoked by such a web-server involve:

- **Vendor Trust.** The web-server application vendor ensures that the system will have high availability and not allow unauthorized and potentially embarrassing changes to the displays, the internally defined links, the data stores supporting the web server application, or any security controls implemented.
- **Administrator Trust.** The web-server administrator built the site and operates it in a fashion that does not allow casual, unauthorized access to other systems on

the hospital network, to the data that the site uses, or to the data a site user creates. This also involves a trust that the operating system will function as designed.

- **Network Trust.** The network administrator ensures that the server site is easily accessible to authorized users, but cannot impinge on other systems except as designed.
- **User Trust.** The authorized system user will not divulge their access authorization (e.g., user IDs and passwords or authorization tokens) to unauthorized users, nor will they make extraordinary attempts to thwart the system design or test the system restrictions to the point of failure.
- **Guest Trust.** The casual visitor (i.e., web surfer) has benign motives in exploring the site.

With this compendium of trusts, Table 1 identifies the risks associated with each trust, along with possible exploits of this risk, and the changes needed to mitigate these risks.

**Table 1 - Web-Server Trusts**  
An examination of the risks, to include methods of exploitation and mitigation.

Risk	Exploit	Mitigation
<b>Vendor Trust</b>		
Unconstrained trust in the vendor and application software exposes the system and makes it vulnerable.	Hostile users could probe the web-server for security holes and weaknesses, such as a missed software update or an incorrect application setting. For many known holes, the scripts for exploiting them can easily be found on hacker, vendor, and security websites. Once the system is compromised, the hostile user can change web pages or links and can also access any data and logs that are on the machine.	Reduce the risk in the application software trust by regularly (daily or weekly) checking vendor and security websites for security problems and software updates.
<b>Administrator Trust</b>		
Unconstrained trust in the system administrator and the operating system could leave the system over-exposed and vulnerable.	Similar to the method of exploiting the vendor trust, a hostile user could probe the system for security holes and run scripts to exploit the holes. Once this web-server trust is compromised, any information that the web-server shares with other systems in the local network can be compromised. The hostile users could even install Trojan horse software that will, for example, allow them to monitor network traffic for user IDs and passwords.	Reduce the risk in the system administrator trust by establishing and implementing policies for the configuration of a web-server and perform scans on a regular basis against the system to verify the configuration and find problems. As for the operating system, regularly check vendor and security websites for security problems and software updates.
<b>Network Trust</b>		
Too much trust between the web-server and the local network exposes the local network to threats from the web-server. At the same time, too little trust frustrates local web-server users.	A hostile user could first compromise the web-server, and then use it to take advantage of the relationship between the web-server and the other hosts on the local network.	Reduce the trust of the web-server by isolating it on a separate network and treating it as an outside machine.

**Table 1 - Web-Server Trusts**  
An examination of the risks, to include methods of exploitation and mitigation.

Risk	Exploit	Mitigation
<b>User Trust</b>  A user's access authorization may be made known. Users are trusted to protect their user ID and passwords, and the system trusts that the user ID and password belong to the assigned user.	If access authorization is divulged, either accidentally or on purpose, the web-server grants the false, perhaps unauthorized, user the same privileges as the authorized user. Also, an authorized user that tests the system restrictions may discover unintentional areas of trust on the web-server. In either case, this results in the risk of data being compromised or changed.	Develop, implement, and enforce a policy on user responsibility of user IDs and passwords. Users must not share user IDs and passwords. Train users in the policy with periodic re-enforcement. Other user-level mitigation options include the use of smart cards, token cards, or biometrics in addition to passwords. Restrict access to sensitive areas on the web-server to only the IP addresses of those that need access.
<b>Guest Trust</b>  The risk incurred with this trust is that everyone is trusted equally. That is, both friendly and malicious sources have equal access to the web-server.	Malicious sources can perform vulnerability probes of the web-server using freely available software from the Internet.	The web-server should be examined not only from the standpoint of casual visitor, but from a malicious one as well. Further examination can be performed with scanning software.

## 2.2 Remotely Administered System Illustration

A distributed information infrastructure frequently includes systems that are installed at the behest of others and are managed remotely. These systems are integrated into the local infrastructure, with the result that they operate with a trust model, which provides complete trust between the remotely administered system and the local infrastructure. Local support personnel may also provide administration or serve as a point-of-contact for users with problems.

At a medical facility, the remotely managed system may simply be an application software package that interfaces with others systems containing patient, insurance, scheduling, staff, pharmaceutical, or supply information. Typically remote administration involves one or more persons at a site remote to the local infrastructure providing support at the application or computer system level or both. The remote administrator has an account with privileges beyond a normal user. This account may be called an administrator or root account. The remote administrator monitors the application/system performance, reviews operating logs, performs updates to the application and/or system, and manages user accounts.

The trusts invoked by such a remote administration capability are listed below. By determining the state of the remotely administered system in terms of the trusts that are either explicitly or implicitly invoked, the associated risks can be identified and managed. Table 2 shows the risks, along with possible exploits, and the changes needed to mitigate these risks.

- **Remote Administrator (RA) Qualifications Trust.** The RA is qualified to perform the administration of the application and system.
- **RA Data Access Trust.** The RA has the proper access level for the data on the system and knows the sensitivity of the data and the restrictions on accessing the data.
- **Remote Connection Trust.** The users and the RA connect to the remotely administered system in a manner that will not compromise the system or those systems connected to it.
- **Operations Trust.** Operations performed by both authorized users and RAs will not adversely impact the other system users or the exchange of data with other systems.
- **User Trust.** The authorized system user will not divulge their access authorization to unauthorized users. Also, the authorized system user will not make attempts to probe other systems in the local network.

Table 2 - Remotely Administered System Trusts  
An examination of the risks, to include methods of exploitation and mitigation.

Risk	Exploit	Mitigation
<b>RA Qualifications Trust</b>		
The remote administrator (RA) may not possess the necessary procedural and technical knowledge to manage and maintain the application and/or system.	Intentional or unintentional actions of the RA in the performance of his/her administration tasks could crash the system or cause disruption while users are accessing it.	Ensure RAs are trained and capable. Reduce the trust given to RAs by establishing policy to keep local administrators informed of the RAs' actions and prevent duplication and/or nullification of efforts. Require transaction or trace logs of all remote administrative sessions so that problems caused by inappropriate actions can be identified and corrected.
<b>RA Data Access Trust</b>		
Unsubstantiated trust that the RA possesses the proper authority to view the data on the system and is aware of the restrictions on accessing the data puts sensitive data at risk of being compromised.	An RA could reveal, or pass along to others, sensitive data located on the remotely administered system, or simply browse the data out of curiosity. The exploitable sensitive information on the remotely administered system includes data input by local personal or that which is automatically gathered from local systems. As a result, other systems may also be put at risk.	Reduce the trust given to the RA by establishing and implementing policy that ensures RAs have the proper level of authority to view any data that they can access and that they are trained on the ethical and legal restrictions on the proper use of the data.
<b>Remote Connection Trust</b>		
Trust in user and RA connectivity, either via external network or dial-in, to the remotely administered system could expose sensitive data to collection devices placed along the communications paths.	Whenever external network connections are used, a hostile user could capture the transactions performed by users and/or the RA by placing a sniffing device at any point in the network where the data passes. If the network traffic is sent in the clear, it can be read directly, thereby easily compromising sensitive data that may include user IDs and passwords. Dial-in connection can provide a backdoor into the remotely administered system and the local network.	Reduce the trust to all connections by allowing connection to the remotely administered system from only selected IP addresses. Use one-time passwords and encryption. Require that all dial-in connections be through a RADIUS [25] complainant system that provides authentication, authorization, and auditing.

Table 2 - Remotely Administered System Trusts  
An examination of the risks, to include methods of exploitation and mitigation.

Risk	Exploit	Mitigation
<b>Operations Trust</b>		
Benign Risk: Normal, day-to-day user operations, and especially RA operations, could interfere with or negatively impact other system usage.	Users could unintentionally cause the system to crash, fail, or lock-up. The RA could take down the system, without warning, while other users or administrators were accessing it. Installing software updates could disrupt not just the target system, but also network traffic.	Train users on proper system usage. Users, and especially the RA, must perform their work so that interference with or impact on other system operations is minimal, maximizing system availability, while minimizing downtime. Establish policy requiring that the RA schedule periodic system maintenance or inform the local support staff administrator when they are working on the machine. Major software updates should be scheduled and coordinated with the local support staff so that network operations can be monitored and the impact minimized.
Malicious Risk: The RA and users could perform unauthorized operations on the remotely administered system.	The RA and users could attempt to connect, without authorization, to other systems (local and external) from the remotely administered system. Software could be loaded on to the system and used to probe local or other remote systems. Sniffing devices could gather local network traffic and capture user IDs, passwords, and other sensitive data.	Isolate the remotely administered system from other parts of the local network. Communication between it and all other systems, local and external, should be limited by a firewall to only that which is absolutely necessary.
<b>User Trust</b>		
Users may fail to protect their user IDs and passwords. The system trusts that the user ID and password belong to the assigned RA or user.	If access authorization is divulged or compromised, either accidentally or on purpose, the remotely administered system will grant a false user the same privileges as the authorized user. Sensitive data can be compromised or changed.	See User Trust for the Web-Server in Table 1.

### 2.3 Analysis of Trust Models

The preceding two illustrations considered what a system trusts (i.e., a trust model) to initiate an analysis of the inherent risks and develop methods of mitigation. These illustrations showed how system operations have anticipated and unanticipated trusts with associated risks that can be exploited. The more complete the trust model, the more complete the analysis, and the more rigorous the mitigating management tools and techniques that can be identified and applied.

Developing and verifying a complete trust model is not trivial. It requires an in-depth understanding of the administrators, users, policies, procedures, computer systems, system dependencies, local area networks, and wide area networks. By understanding the composition of the trust model and the risks that result, the trust model can be narrowed, and the associated risks managed. The concept of a trust model not only provides a realistic perspective of which users/systems have access to the targeted system, but it also allows a deeper understanding of trust in terms of access and permissions. For instance, permission to access and read within a file system is one level of trust, while permission to access and write is a deeper level of trust. The next section develops the context for a trust model of interdependent systems.

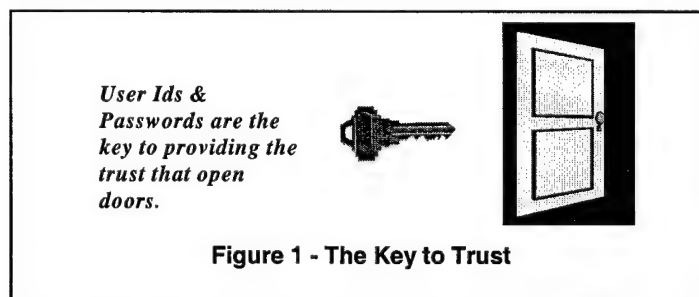
### 3. AN OVERVIEW OF TRUST MODELS IN SYSTEM AND NETWORK OPERATIONS

Modern computer systems rely on trust relationships to operate, trusting the system administrator, the privileged system users, and the interconnected system, but with these trusts come the risks that the trusts will be violated. Garfinkel and Spafford [1] and others have written about trust, to include the trust between people and organizations. While that level of trust is fundamental, it is too broad in scope for the purposes of this work. We concentrate on trust relationships specific to computer systems and network operations, starting with the trust that a user is granted or denied on a computer system then expanding to cover not only the user's computer system but also the interface with other systems and networks.

#### 3.1 Computer System Trusts

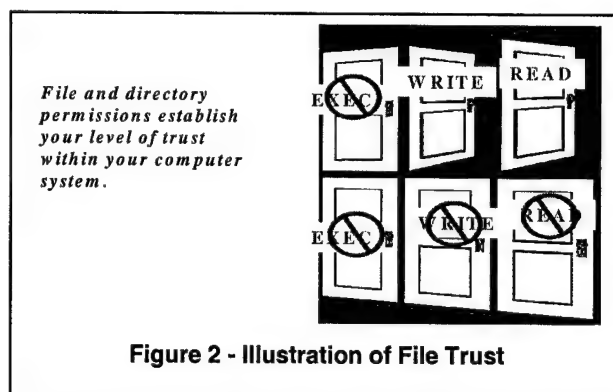
In this section, we build a framework for computer system trust that starts simple and grows more complex. That is, the discussion starts with trust relationships in terms of the user simply gaining access to the system and builds to more complex situations in the discussion of operating systems and application software.

**3.1.1 User ID and Password Trusts.** When a user is given a computer account, they are assigned a user ID and password and they inherit certain trusts that go with the account. The computer will not extend trust to the user (i.e., grant access) until the correct user ID and password are provided, as illustrated in Figure 1. The computer has no other way of identifying the user. The user ID and password are like a key that will open the door of the user's system.



The risk associated with user IDs and password is that the user ID and password can be lost, stolen, or guessed, then exploited. Given a valid user ID and password, the computer will open the door to the unauthorized user the same way it would to the authorized user. Mitigating this risk requires policies and sound practices covering who is authorized have an account, how the user IDs and passwords are transferred without risk of compromise, composition and expiration of passwords, and user protection of their IDs and passwords. The first line of defense in computer system trust is the protection of user IDs and passwords.

**3.1.2 Directory and File Trusts.** Once the user has an account on a computer system, the user may only access certain directories and have certain privileges with the files in these directories. Directories and files generally have at least three trusts associated with them: permission to execute, permission to write, and permission to read. These levels of trust are generally repeated for three user populations: for the owner, for groups, and for everyone (a.k.a. world). The ability to perform these permissions can be represented by an open door, and the inability to perform each can be presented by a closed door, as illustrated in Figure 2.



The system administrator gives a user permissions when an account is created, normally in accordance with some authorized user profile. These initial permissions establish the trust relationship between the user and members of the user's group, all other users, and the system. A user's permissions can be amended over time by including them in additional groups. The user then inherits the permissions granted to the groups. Operating systems may also apply default settings to file permissions, and within the limitations of one's level of access, an individual user may also grant permissions between themselves and other users or groups of users. When user files and directories are configured with world read, write, or execute permissions, the user trusts all users, those known to the file owner and those not known.

To exploit the trusts associated with intentional and unintentional (to include inherited) granting of permissions, an unauthorized user (e.g., a user that may be malicious, mischievous, error-prone, etc.) can take advantage of the permission setting that allows access. One can do this by simply searching for user and system files that have world permissions or accessible group permissions.

Administrators must use caution when applying permissions. If, for example, system files and directories are configured with world write permissions, then every user is trusted. This model of complete trust can result in catastrophe: With such access to the file system, malicious users are capable of deleting or replacing system files. They could even manipulate system level commands or install Trojan commands (i.e., user programs disguised as system commands). As a result the system's integrity is left questionable. The system could either crash unexpectedly or not behave as intended or perhaps worst, the system may seem operate normally while Trojan commands relay information, unbeknownst to the administrator or users, to the

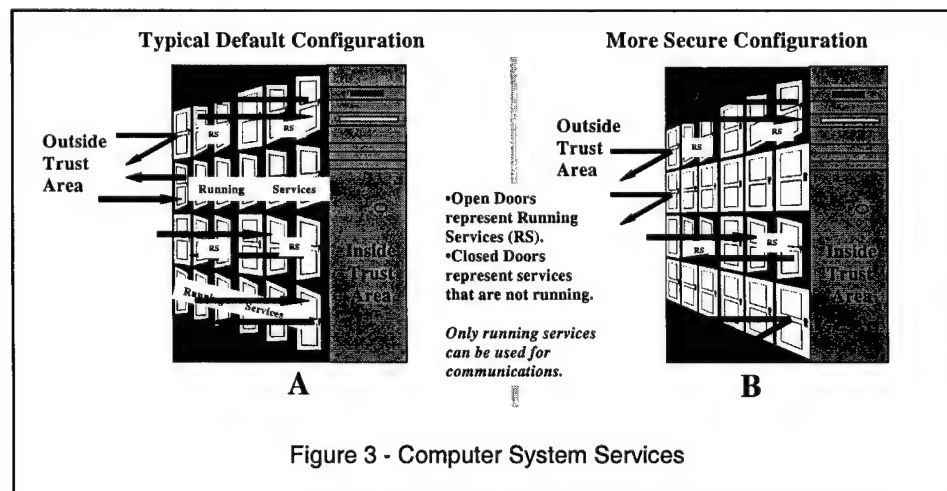


malicious user. Having world writable permissions on a large number of system files is like handing the keys to the kingdom to any passerby. Some operating systems are better than others in having a minimal number of world writable file permissions established from default settings.

To mitigate the risk that an unauthorized user, by accident, mischief, or malice, can change or delete information, file and directory permissions must be carefully granted. In granting permissions to other users, the user is providing them with the trust to access files and directories that they would not otherwise have permission to access. The system administrator can improve system security by selectively making system files and directories read only and execute only. References [1-4] provide guidance in this area and provide a number of additional ways to increase computer system security.

**3.1.3 Operating System Trust.** Operating system trust is multifaceted because an operating system performs all the diverse, yet necessary, activities that make a computer function properly in addition to handling tasks like network communications, data sharing, and user logins. Since the operating system handles user logins, directory and file permissions, and application program interfacing, the trusts described in Sections 3.1.1, 3.1.2, and 3.1.4 also apply to the operating system. The focus of this section, therefore, is the trust of another area covered by the operating system - that of computer "services." These services are programs that communicate (talk and listen) using IP protocols and ports. When the services are activated, they wait for input from like services and will respond to requests. Common services include sendmail, Telnet, FTP, and HTTP. Some services have complete trust and will respond to any requests; other services have less trust and require authentication before honoring a request [17, 23].

When a computer is initially configured, many services are enabled by default, as shown in Figure 3A. TFTP, echo, and chargen are three such services with complete trust that are frequently enabled by default on Unix systems. On many Windows NT systems, web-server and FTP services are enabled by default. All too often, the user or even the administrator is unaware that these services are running.





The risk is that these services, running by default or explicitly enabled, can be exploited to attack a computer system. They provide paths into the computer. A system can be probed for potentially exploitable services using software readily available from the Internet. From this probe, one can frequently gather enough information to determine the type of operating system (Unix, Windows, Mac) in use, the version of the operating system, and what services are active [18]. Knowing this information a potential intruder can target known security flaws. Multiple hacker web sites, and even security related web sites (e.g., the CERT® [19] and National Infrastructure Protection Center [20] sites, to list just a couple), provide a wealth of information about known, exploitable flaws. The hacker sites may even provide the software that can exploit the flaw.

The simplest way to mitigate these risks is to minimize the services running on the computer. While restricting services to the minimum needed will reduce the potential trusts available for a computer system, it will improve the system in terms of security, as shown in Figure 3B. One of the problems in taking a minimalist approach to service is that the system administrator may not know what services are required. References [1-4, 14, 15] provide guidance for eliminating services that are not required. As the need for services changes, they can be added or deleted with minimal effort. The cost is the system administration time to override defaults during setup, to adjust to changing users' needs, and to keep abreast of new and potential exploits. Such costs are insignificant when compared with the potential savings in time and computer resources by mitigating the risks.

Installing a Border Guard type of software provides additional protection for a computer system. This software wraps the computer in a protected perimeter and allows the trust requirement for each computer to be established separately. TCP-Wrappers [6] is a well-known program offering this protection in a Unix environment. It allows the system administrator to establish permissions (based on source address, time, etc.) for each service, thereby acting as a mini firewall. The Windows (95, 98, NT, and 2000) environment also has a number of choices including: BlackIce Defender [7], BackOfficer Friendly [8], Nuke Nabber [9], and ZoneAlarm [10]. These programs provide a degree of local intrusion detection and the ability to block potentially dangerous sites from accessing a computer.

**3.1.4 Application Program Trust.** The concept of trust carries over to application programs. A user may not be able to execute an application program because the user's permissions do not include execute for the application. In this way, the computer system file permissions regulate the level of trust. In other application programs, the program itself determines the trust for users by requiring that a user provide access authentication, such as another user ID and password. This is like two doors in series; one must have the keys to unlock both doors before gaining access. These application program user IDs and passwords are not necessarily the same as the user's computer system user ID and password, and while it is common to use the same user ID, the system password should not be used.

Application programs may also have different levels of user trust. Normally, the application program administrator will have a different level of trust than a user since the administrator must accomplish administrative activities that would be hazardous to allow a user to do. Users may actually be assigned an access based on their roles (i.e., doctor, nurse, insurance filer, etc). Trust based on roles is controlled by the application program and is determined mainly by the user ID and password. If a person has multiple roles, the program may verify which role they are exercising and grant privileges based on that role.

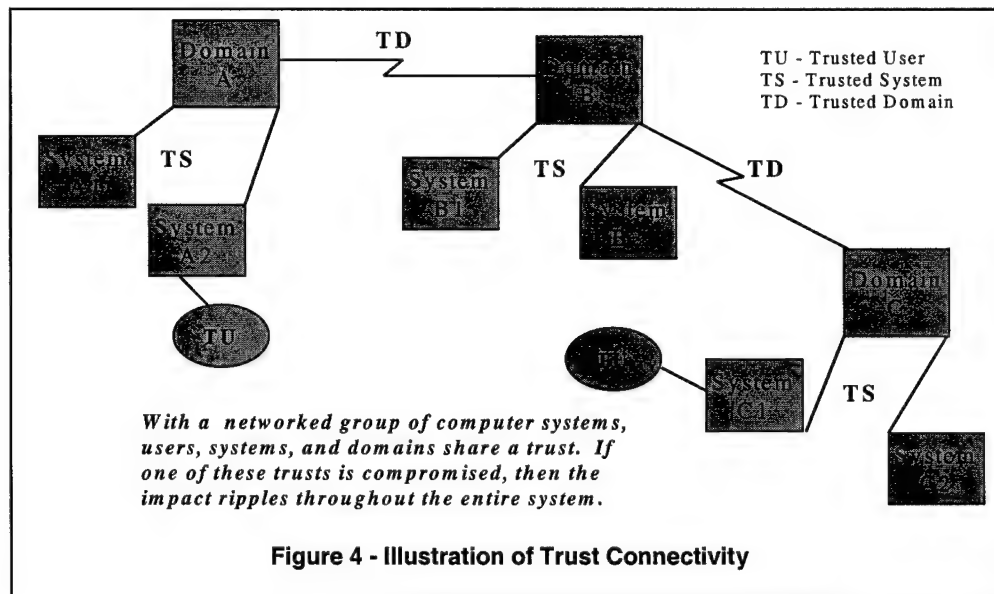
Each application program has a trust relationship between itself, the operating system, and other application programs. Application programs may, for instance, interface with each other so that data can be shared among them. These relationships are determined when the system or application administrator configures the application program. The application program should be configured so that only authorized users have access to the shared data. This activity involves trusting the administrator to properly configure the program.

To exploit security weaknesses in an application program, a malicious, unauthorized user could try a whole array of hacker toolkits. Such toolkits are readily available for download from the Internet. They are capable of identifying and even exploiting a whole range of security weaknesses, from inappropriately assigned file permissions to security holes in the application program.

To mitigate the risks involved with application programs, only users that need to use the application should be authorized to access and run the application. Additionally, user privileges within the application program should reflect the user's role and establish limitations accordingly. The application administrator should also minimize the trust relationships among the application programs and the trust with other systems. Finally, the application administrator must regularly (daily or weekly) check the vendor websites for software updates that impact system and application security, then apply the updates in a timely manner.

### **3.2 Trust Between Computer Systems**

The versatility of today's modern computer network comes from the ability of computers to share data across local and wide area networks. This versatility involves trusting other computer systems and the networks that connect them, and therefore, it incurs risk. As discussed in Section 3.1, a user account on a computer system inherits certain levels of trust on that system. There are also levels of trust between computer systems. In some cases the system-to-system trust extends to the user, and in some cases the trust is limited just between the systems. These extensions of trust may provide access to software, file systems, and email systems, or they may provide a single user ID and password on multiple systems. Figure 4 shows a representation of the trust interconnects between computers, domains, and networks, wherein a trusted user on System A1 is also a trusted user on System C1.



This ability to have trust between systems is very important in a networked computer environment. It allows machines to be dedicated to certain functions so that it is not necessary to duplicate functions across the network. There are also levels of trust between systems. For example, system database inquiries may be made between two systems, but the database can only be updated by one of the systems. A group of computer systems that share a common trust is usually referred to as a domain. There may also be trust relationships between domains, as shown in Figure 4. In most cases, the network administrator must explicitly define trust among domains.

This interconnection of systems and domains, already with the network of trusts created, can be referred to as "connectivity" trust. That is, if a system is trusted, then anything that system trusts is also trusted. The risks with connectivity trust are that unknown and unpredictable paths exist and are potentially available as trusted paths. Users may be able to take advantage of the trust on their system to access other systems to which they would normally not have access. One of network security's greatest challenges is that a system can be attacked indirectly through computer systems that it trusts. For examples of this vulnerability see Garfinkel and Spafford's [1] discussion of Unix's `.rhosts` and `hosts.equiv` files.

Mitigation of the risks associated with proliferation of potentially unwarranted trusted paths involves protection of not only the local network and its computers, but protection against the possible compromise of systems that are connected to the local network. Network risks can be mitigated by reducing the trusts that the local computers have for each other. If a trust relationship is needed between two computer systems, the relationship should be limited to only what is needed to accomplish the required tasks. If bi-directional trust is not needed, then use only a one-way trust. Wide area network risks can be mitigated by reducing the trust that the local network has for the wide area network. The same principles apply; that is, minimize the trust relationship to only that which is required.

### 3.3 Border Guard Trust

**3.3.1 The In's and Out's of Trust.** The ability to enter and leave a country is an important trust, granted by a government, based on some criteria. Granting or denying this trust is determined by whether a person matches the established criteria. A border can be represented as the entry and exit point of a country, as shown in Figure 5. In this scenario, each door has the potential to be opened or closed, and

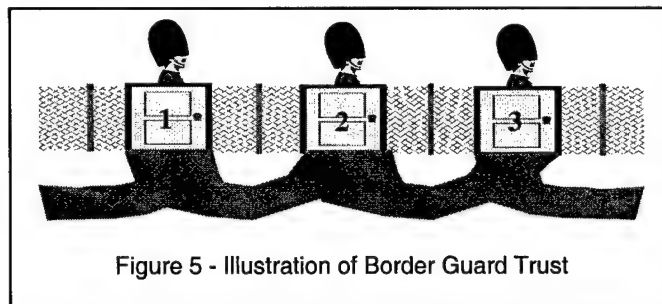


Figure 5 - Illustration of Border Guard Trust

there can be rules for entering as well as leaving the country. Each guard has a separate set of criteria and specific actions to be taken if the person matches those criteria. If there is a match, for example, the guard has two possible actions: either the door could be opened, allowing the person entry, or the door could remain closed, denying the person entry. If there is not a match, the person must go to the next door. Doors must be visited sequentially, and the first that matches the condition applies (i.e., no more doors will be visited once a match is made).

Shown below is a simple example of criteria matching using a generic Case Statement:

```

CASE Incoming_Person OF
  Door_1: (Tourist AND Walking): Deny
  Door_2: Tourist: Permit
  Door_3: (Professor AND Dollars >= $1000): Permit
  Door_4: Professor: Deny
  Last_Door: Everyone: Permit
END

```

In this example, the incoming person (assuming a male) is first tested at Door\_1 to determine if he is both a tourist and that he is walking. (Notice the use of the logical AND.) If there is a match, then Door\_1 remains closed and entry is denied, but since there was a match at Door\_1, no other doors are visited. If there is no match, he goes to Door\_2 to determine if he is just a tourist. If there is a match, then Door\_2 is opened and access is allowed. If there is no match, he goes to Door\_3 to determine if he is a professor with at least \$1000. If there is a match, then Door\_3 is opened and access is allowed. If there is no match, he goes to Door\_4 to determine if he is just a professor. If there is a match, then Door\_4 remains closed and access is denied. If there is no match, he goes to the last door. He is tested at the last door to determine if he is a member of the group that includes everyone. Since every person is a member

of this group, the door is opened and access is allowed for all who make it to this door.

In analysis, we notice that this is a very trusting country: Only walking tourists and professors that do not have at least \$1000 are denied access. To keep more people out, doors with more restrictive criteria would be required. It is also worth re-iterating that the doors are checked in sequential order, and when a match is made, no more doors are checked. For example, if the incoming person is both a tourist and a professor, he would meet the criteria of Door\_2, he would be granted access at that point, and he would not be checked at Door\_3 or Door\_4.

A different set of criteria may be applied to those people leaving the country to determine whether or not they will be permitted to do so. Here is another example of criteria matching using a Case Statement:

```
CASE Outgoing_Person OF
  Door_1: Tourist: Permit
  Door_2: Doctor: Deny
  Door_3: Professor: Deny
  Last_Door: Everyone: Permit
END
```

In the case of outgoing people, only doctors and professors, who are not tourists, are denied the freedom to leave. Everyone else is permitted to leave. The trusts embodied with both of the examples given above are broad because the last door in both the incoming and outgoing cases is a "permit." This means that if no other criteria match, then let them through.

To apply this to computers and networks, a method is needed to establish the criteria and the entities to be evaluated must be determined. Rules will be used to establish the criteria. These rules will be based on Internet Protocol (IP), the dominant protocol in computer networking. IP defines a packet that the transmitted information fits inside. The packet identifies a source and destination address, a set of protocols (TCP, UDP, ICMP, etc.), and ports. These elements (IP address, protocol, and port) provide a means for applying a rule (criteria) to permit or deny packet communications. As stipulated in the case statements above, rules are evaluated sequentially, and the first rule that matches the conditions applies. The following is an example of criteria evaluation and resultant action applicable to incoming packets:

```
CASE Incoming_Packet OF
  Rule_1: (From any address AND To web-server xyz1 AND port 80):
  Deny
  Rule_2: (From any address AND To web-server xyz2): Permit
  Rule_3: (From 123.1.1.1 AND To 144.2.2.2 AND TFTP): Permit
  Rule_4: (From any address AND To any address AND TFTP): Deny
  Last_Rule: (From any address AND To any address
              AND any protocol): Permit
END
```

Once again, a different set of criteria can be used to evaluate outgoing packets to determine whether or not they are permitted to leave:

```

CASE Outgoing_Packets OF
  Rule_1: (From any address AND To any address AND HTTP): Permit
  Rule_2: (From 144.2.2.2 AND To any address): Deny
  Rule_3: (From any address AND To 123.1.1.1): Deny
  Last_Rule: (From any address AND To any address
              AND any protocol): Permit
END

```

Again, the trusts embodied with these criteria are broad because the last rule in both the incoming and outgoing cases is “permit.” This means that if no other criteria match, then let them through. This situation is described in more detail in Section 3.3.2 below.

Firewalls are often used to act as Border Guards. Packet filters can be applied to check the criteria and enforce the rules. References [5, 11-13, 16] provide detailed information in using firewalls as Border Guards.

**3.3.2 Trust Everything that Is Not Specified**. A “trust everything” rule permits (trusts) any source address going to any destination address using any protocol on any port. As illustrated for incoming packets in Figure 6, when a packet comes to a door (i.e., rule), the packet is evaluated against that rule. If the packet matches the rule, it is either trusted (the door is opened) or denied (the door remains closed) based on what the rule determines. If the door is opened, the packet that matches the rule is permitted inside. Packets are evaluated against each rule sequentially until there is a match. Once a match is made no other doors are checked. The large door (at the lower right) represents the last rule. If no previous rules applied, the last rule does; it can be stated as:

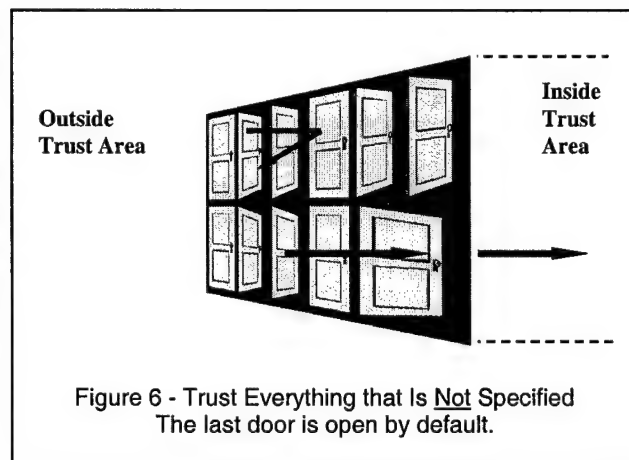
```

Last_Rule: (From any address AND To any address AND any
            protocol): Permit

```

In effect, the last door is always open and the Border Guard is therefore termed a “Trust everything that is not specified” type.

In some ways, this configuration is rather simple to manage because everything is trusted (allowed). However, when it becomes necessary to increase the level of security in the Border Guard against the outside trust area, this method is not very



effective. You can, with difficulty, protect against the known threats, but with this configuration you cannot protect against the unknown threats.

No longer can system administrators worry about security simply in terms of what's on the outside trying to get in. Web browsing and FTP provide the means for passing information about users and disk contents to outside entities (individuals, companies, governments, etc.). Trojan software can also communicate with outside systems. Thus, over the last several years, the need for two-way trust has developed, and in many cases, Border Guards can handle both incoming and outgoing packets, as shown in Figure 7. Border Guards that check only incoming traffic, but not outgoing traffic, are providing only partial coverage of corporate assets. Two-way checking is always better than one.

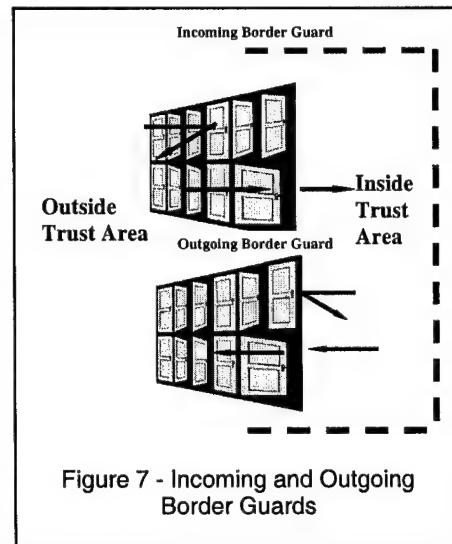


Figure 7 - Incoming and Outgoing Border Guards

**3.3.3 Trust Only What Is Specified.** Figure 8 depicts another Border Guard with many doors (rules). The evaluation of the rules is the same as in Section 3.3.2. The difference here is with respect to the last rule. In this case, the last rule would be stated as:

```
Last_Rule: (From any address AND To any address AND any
            protocol): Deny
```

In other words, the last rule states that for any IP address, any protocol, and any port, do not trust them. The last door remains closed, and the Border Guard is referred to as a "Trust only what is specified" type. As mentioned in Section 3.3.2 (Figure 7), this type of Border Guard can be applied to both the incoming and outgoing packets.

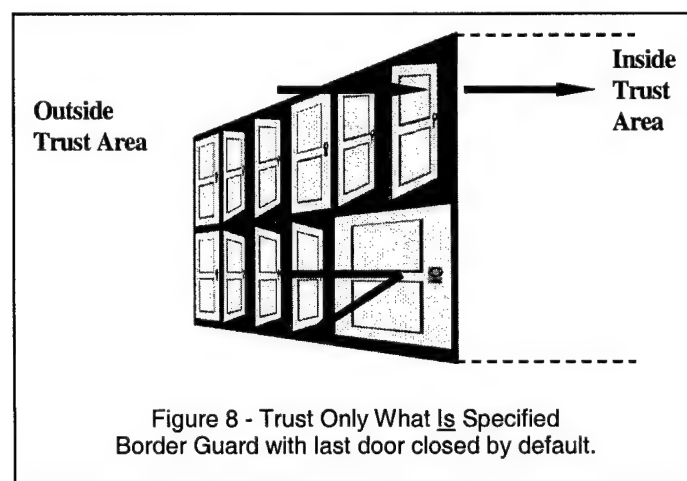


Figure 8 - Trust Only What Is Specified Border Guard with last door closed by default.



There is a large difference between the "Trust only what is specified" type and its opposite, the "Trust everything that is not specified," which was discussed in Section 3.3.2. Trusting only what is specified offers the least amount of trust, but inherently it offers the most security protection. This configuration also requires more management because a rule must be created for each instance that the Border Guard should allow packets to be trusted. The stricter the rules (e.g., only the required protocol allowed from an IP address vs. all protocols), the less trust and the more protection is offered. At first glance, this strategy may seem great; however, in environments that are inherently more open (e.g., universities, research labs, etc.), this strategy may not work well. In such cases, Border Guards with limited trust should be used inside to protect sensitive areas.

## 4. APPLICATION OF TRUST MODELS

A trust model describes the trust relationships that a computer system or network possesses so users and administrators can understand and hopefully minimize the associated risks and vulnerabilities. This section provides guidance for developing a trust model by revisiting the web-server and remotely administered system illustrations and applying the trust relationships developed in Section 3. The resulting trust statements are grouped to address four major categories of trust: access, data, operations, and communications. These categories address and incorporate the trust relationships developed in Section 3.

### 4.1 Development of Trust Models

In appearance, the trust model is simply composed of a set of trust statements. Each of these statements tells the following: what is the entity trusted, what action is being trusted, and who/what is trusted to perform that action or what is necessary to perform the action. The format, which is similar to the format of an English sentence (i.e., subject - verb - object), is given by:

<entity of trust> <action being trusted> <who or what is trusted>

Where entity = computer, network, or person  
person = administrator or user  
administrator = system or application administrator  
user = all users, selected users, benign users  
or malicious users

Development of the trust statements involves identifying and understanding the risks and vulnerabilities, then forming trust solutions, based on sound justifications, to address the risks and vulnerabilities. Within each trust category (access, data, operations, and communications), attempt to define broad, high-level trust statements first. Accordingly, high-level solutions are associated with the high-level trust statements. As the trust statements become more specific, the solutions should also become more specific. A trust statement may not be resolved with a single solution; at the same time, a solution may resolve more than one statement. The incurred



redundancy may be acceptable, and even desired, if it results in an additional layer of protection.

The more completely that a trust model can be defined, the greater awareness one gains concerning the risks incurred by granting trust in a potentially threatening environment. Despite the ever-growing number of threats that a system may face, a trust model is not necessarily bound to advocate or force a solution for each risk. Once defined, the trust model should instead be used as a tool to better understand and minimize the risks, identifying those that can be mitigated by implementing a solution and those that must be accepted.

## **4.2 Application to the Web-Server System**

In Section 2.1, a web-server illustration was presented that included a list of trust relationships, the risks associated with each trust, possible exploitation methods, and risk mitigation actions. This section will define a new set of trust statements that have reduced risks and describe ways to implement these trusts at the computer, application, and network level. The basic format for developing the trust model, which consists of the trust statement followed by its trust solution and justification, is illustrated below for the first web-server trust statement, which happens to address access trust.

**Trust Statement:** The web-server provides access to information for the public.

**Trust Solution:** Use a dedicated machine for the web-server. Restrict/limit the operating system services, other applications, and connections to other machines.

**Justification:** It is important that the web-server not run other server software or services, such as domain name server (DNS), email, and simple network management protocol (SNMP). Since the web-server is publicly known and accessible, it provides an attractive target for those trying to exploit a system. Running superfluous software provides additional avenues of attack and increases the impact of a compromise. If the web-server is compromised, and that same computer is also running DNS and/or SNMP, the entire site could be at risk.

To minimize the confusion that would result from a rather lengthy list of all the trust statements, along with each statement's associated trust solution and justification, located in this section, the complete development of the web-server trust model is provided in Appendix A.

**4.2.1 Border Guard Layout and Architecture.** The treatment of Border Guards requires some additional explanation. Border Guards, implemented with either a firewall or router, can be used to protect a web-server. In a typical architecture, upon which we will expand, the demarcation BG (DBG) is a router and the site BG (SBG) is a firewall. (The DBG is located at the extreme point where a site connects to an external network.) The trust solutions for the web-server trust model depend on which architecture is implemented and what each of the Border Guard is protecting. A number of architectures are available; we will focus on three that can be reasonably effective.

**Architecture 1.** In Figure 9, the web-server is in the so-called de-militarized zone (DMZ), located between the DBG and the SBG. With this physical architecture, there are a number of possible trust settings for both the DBG and the SBG. For example, if the DBG were fully trusting (especially of incoming traffic), the web-server would be severely at risk and highly vulnerable to attacks from the external network. Thus, for this architecture to be effective, the DBG must minimize the trust of the external network. The SBG also protects the site's internal network from external attacks, to include attacks from the web-server, as well as any internal network attacks on the web-server. If the SBG does not minimize the trust of the external network and the web-server, the internal network is at risk.

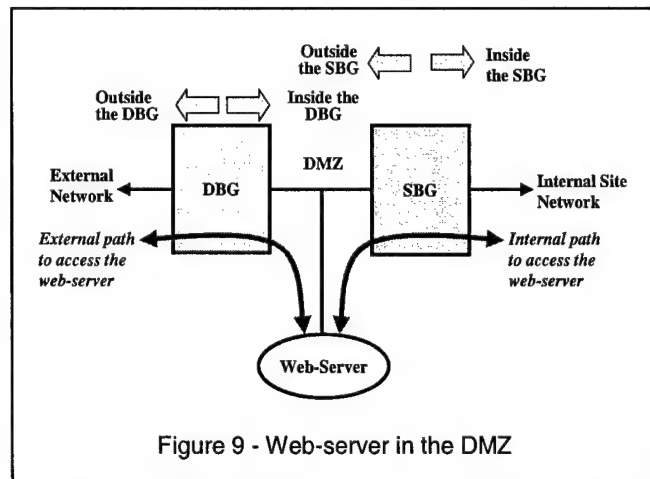


Figure 9 - Web-server in the DMZ

**Architecture 2.** Another architecture is shown in Figure 10A. In this architecture both the DBG and the SBG provide protection for the site's internal network, to include the web-server, from the external network. This architecture implies maximum trust between the site's internal network and the web-server. Unfortunately, if the web-server is compromised, the other computer systems on the internal site network are unprotected from the web-server; thus, it can be used to mount an attack against those computers. Also, if the web-server receives large amounts of traffic from the external network, the internal site network may be impacted.

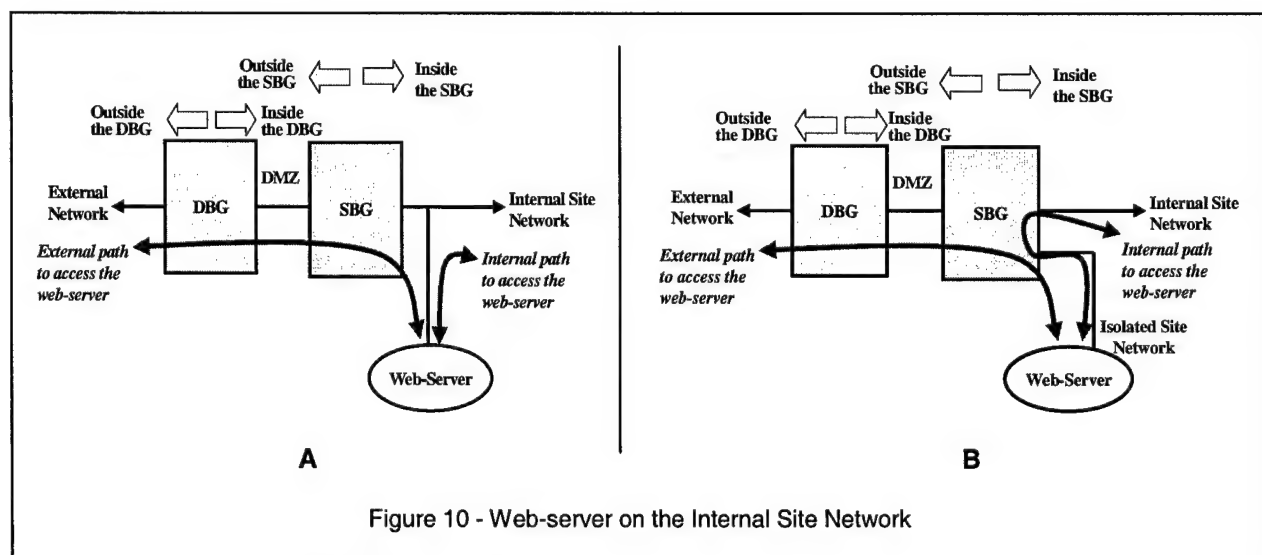


Figure 10 - Web-server on the Internal Site Network

**Architecture 3.** In Figure 10B, both the DBG and the SBG again provide protection for the site's internal network and the web-server. However, the web-server is now on a different interface, thereby providing isolation from the site's internal network. Since each interface can have its own set of rules, there can be minimum trust between the web-server and the internal network. By isolating the web-server, not only is the internal site network protected from an attack mounted from the web-server, but the impact on the internal site network traffic is also reduced compared to Architecture 2.

From the three architectures described above, four configurations will be used as points of reference for creating and adding to the trust model. These configurations and their impact on the web-server and site internal network are described in Table 3.

Table 3 - Border Guard Configurations

Configuration	Impact on the Web-Server and the Site Internal Network
1. Architecture 1 with full trust at the DBG and minimal trust at the SBG.	The web-server is highly vulnerable to attacks from the external network. If compromised, the web-server could become a gateway to other computers external to the DBG. Snooping software on the web-server could intercept communications between the external and internal network and make any clear text transmission vulnerable [17].
2. Architecture 1 with minimal trust at both the DBG and the SBG.	The web-server is well protected from attack from both the external and the site internal network. The impact of web-server traffic is minimal on internal site network traffic. Again, snooping software on the web-server could intercept communications between the external and internal network.
3. Architecture 2 with minimal trust at both the DBG and the SBG.	The web-server is protected from attack from the external network, but not an attack from the internal site network. A high volume of web-server traffic could have a large impact on the internal site network. If compromised, the web-server could become a gateway to other computers internal to the SBG. Snooping software on the web-server could intercept communications on the internal site network and make any clear text transmission vulnerable.
4. Architecture 3 with minimal trust at both the DBG and the SBG.	The web-server is protected from attack from both the external and the site internal network. The impact of web-server traffic is minimal on internal network. If compromised, the web-server would be limited in providing access to other computers. Snooping software could only intercept communications to/from the web-server.

**4.2.2 The Web-Server Trust Model.** The Web-Server Trust Model, shown in Table 4, consists of trust statements for the web-server system, grouped in terms of access, data, operations, and communications trusts. While these statements define the full trust model, the detailed, background development of this trust model, consisting of the trust statements, trust solutions, and justifications is found at Appendix A.

**Table 4 - Web-Server Trust Model**  
 These Trust Statements constitute a trust model for a web-server.

**1.0 Access Trust**

- 1.1 The web-server provides access to information for the public. (Addresses the trust relationships discussed in Section 3.1.)<sup>1</sup>
- 1.2 The web-server operating system and web applications limit access for actions such as updating web pages to only administrators and specific users with a need. (3.1.1, 3.1.3, 3.1.4)
- 1.3 The web-server operating system limits non-web access to only authorized users. (3.1.3)
- 1.4 The web-server application program grants only selected users administrative access. (3.1.4)
- 1.5 The web-server application program grants access to sensitive data on the web-server to only selected users. (3.1.1, 3.1.4)

**2.0 Data Trust**

- 2.1 The web-server may not directly access the file system on other systems. (3.1.2, 3.2)
- 2.2 Other systems may not directly access the file system on the web-server. (3.1.2, 3.2)
- 2.3 Web-server data access is limited to only those users that need access. (3.1.2)
- 2.4 The web-server application program protects web-server data from accidental release. (3.1.2, 3.1.4)
- 2.5 The web-server application program protects database data from change, deletion, and compromise. (3.1.2, 3.1.4, 3.2)

**3.0 Operations Trust**

- 3.1 The web-server operating system functions in a well-known and proper manner. (3.1.3)
- 3.2 The web-server operating system minimizes the exposure from unused services. (3.1.2, 3.1.3)
- 3.3 The web-server operating system limits denial of service attacks. (3.1.2, 3.1.3)
- 3.4 The web-server operating system has installed only those programs needed for web-server operation. (3.1.3)
- 3.5 The web-server operating system's integrity is protected from unauthorized changes. (3.1.2, 3.1.3)
- 3.6 The web-server application program is protected from known malicious sites. (3.1.4, 3.2)
- 3.7 The web-server application program functions in a well-known and proper manner. (3.1.4)
- 3.8 The web-server application helper programs function in a well-known and proper manner. (3.1.4)

**4.0 Communications Trust<sup>2</sup>**

- 1.2 The web-server operating system and web applications limit access for actions such as updating web pages to only administrators and specific users with a need. (3.2, 3.3)
- 2.1 The web-server may not directly access the file system on other systems. (3.2, 3.3)
- 2.2 Other systems may not directly access the file system on the web-server. (3.2, 3.3)
- 4.1 The web-server will limit damage if compromised. (3.2, 3.3)
- 4.2 Communication is limited to only that which is necessary with other local site systems. (3.2, 3.3)
- 4.3 The Border Guards minimize the exposure from unused services. (3.2, 3.3)
- 4.4 The Border Guards limit denial of service attacks. (3.2, 3.3)
- 4.5 The Border Guards protect the web-server from known malicious sites. (3.2, 3.3)

**Notes:**

- 1. Each trust statement addresses at least one of the trust relationships discussed in Section 3. For reference, the sub-sections are restated below:

- 3.1 = Computer System Trusts

- 3.1.1 = User ID and Password Trusts

- 3.1.3 = Operating System Trust

- 3.2 = Trust Between Computer Systems

- 3.3 = Border Guard Trust

- 3.1.2 = Directory and File Trusts

- 3.1.4 = Application Program Trust

- 2. Some Trust Statements are restated for the Communications Trust category since new Trust Solutions apply (see Appendix A for details).

### 4.3 Application to the Remotely Administered System

The Web-Server Trust Model now becomes the basis for the Remotely Administered System Trust Model (and conceivably, a number of other trust models as well). Most of the trust statements in Table 5, while tailored to remotely administered systems, are clearly based upon the statements in Table 4. The Border Guard architectures differ in this application because now only Architectures 2 and 3, described in Section 4.2.1, are viable. Despite the fact that much of the Web-Server Trust Model is applicable to the remotely administered system, for completeness, the remotely administered system does require some additional trust statements, solutions, and justifications, two of which are developed below:

**Trust Statement:** The remote administrator possesses the clearance level required to access the sensitive data on the system.

**Trust Solution:** Establish policy that requires that all system administrators, including remote administrators, have the proper clearance level and training to access sensitive data.

**Justification:** The requirement of data confidentiality is the same whether the administrator is local or remote.

**Trust Statement:** The remote administrator connects to the remote system in a manner that will not compromise the remote system or the systems connected to it.

**Trust Solution:** Applying the same approach used for the isolated network in Architecture 3, Figure 10B, one can minimize the trust between the remotely administered system and systems on the site's internal network. Connectivity to the remote site and to other site systems can be restricted to selected IP addresses and ports. The use of SSH or IPSEC will protect all communications with the remotely administered system, thereby limiting the effectiveness of snooping. Architecture 2 could also be applied, but it exposes the site's internal network to considerably more risk. On Unix computers, the program *sudo* can be used to provide tailored access to operating system commands [24]. It also provides for the logging of each operating system command. If dial-in access is used, require that access be obtained from a RADIUS [25] compliant system.

**Justification:** Since administration requires execution of privileged commands, it is imperative that the user's ID and password not be compromised. Using the layered approach of an isolated system with IP address and port restrictions, secure communications, and command logging, minimizes the trust and protects against a single point of failure.

Table 5 - Remotely Administered System Trust Model  
These Trust Statements constitute a trust model for a remotely administered system (RAS).

#### 1.0 Access Trust

- 1.1 The RAS provides access for authorized users. (3.1)<sup>1</sup>
- 1.2 The remote administrator possesses the clearance level required to access the sensitive data on the system. (3.1.1, 3.1.2)
- 1.3 The RAS operating system and applications limit access for actions such as updating application data to only administrators and specific users with a need. (3.1.1, 3.1.3, 3.1.4)
- 1.4 The RAS operating system limits Internet access to only authorized users. (3.1.3)
- 1.5 The RAS application program grants only selected users administrative access. (3.1.4)
- 1.6 The RAS application program grants access to sensitive data on the RAS to only selected users. (3.1.1, 3.1.4)

Table 5 - Remotely Administered System Trust Model  
 These Trust Statements constitute a trust model for a remotely administered system (RAS).

## 2.0 Data Trust

- 2.1 The RAS may not directly access the file system on other systems. (3.1.2, 3.2)
- 2.2 Other systems may not directly access the file system on the RAS. (3.1.2, 3.2)
- 2.3 RAS data access is limited to only those users that need access. (3.1.2)
- 2.4 The RAS application program protects RAS data from accidental release. (3.1.2, 3.1.4)
- 2.5 The RAS application program protects database data from change, deletion, and compromise. (3.1.2, 3.1.4, 3.2)

## 3.0 Operations Trust

- 3.1 The RAS operating system functions in a well-known and proper manner. (3.1.3)
- 3.2 The RAS operating system minimizes the exposure from unused services. (3.1.2, 3.1.3)
- 3.3 The RAS operating system limits denial of service attacks. (3.1.2, 3.1.3)
- 3.4 The RAS operating system has installed only those programs needed for RAS operation. (3.1.3)
- 3.5 The RAS operating system's integrity is protected from unauthorized changes. (3.1.2, 3.1.3)
- 3.6 The RAS application program is protected from known malicious sites. (3.1.4, 3.2)
- 3.7 The RAS application program functions in a well-known and proper manner. (3.1.4)
- 3.8 The RAS application helper programs function in a well-known and proper manner. (3.1.4)

## 4.0 Communications Trust<sup>2</sup>

- 1.2 The RAS operating system and applications limit access for actions such as updating application data to only administrators and specific users with a need. (3.2, 3.3)
- 2.1 The RAS may not directly access the file system on other systems. (3.2, 3.3)
- 2.2 Other systems may not directly access the file system on the RAS. (3.2, 3.3)
- 4.1 The RAS will limit damage if compromised. (3.2, 3.3)
- 4.2 The remote administrator connects to the remote system in a manner that will not compromise the remote system or the systems connected to it. (3.1.2, 3.2, 3.3)
- 4.3 Communication is limited to only that which is necessary with other local site systems. (3.2, 3.3)
- 4.4 The Border Guards minimize the exposure from unused services. (3.2, 3.3)
- 4.5 The Border Guards limit denial of service attacks. (3.2, 3.3)
- 4.6 The Border Guards protect the RAS from known malicious sites. (3.2, 3.3)

### Notes:

1. Each trust statement addresses at least one of the trust relationships discussed in Section 3. For reference, the sub-sections are restated below:
 

3.1 = Computer System Trusts	3.1.2 = Directory and File Trusts
3.1.1 = User ID and Password Trusts	3.1.4 = Application Program Trust
3.1.3 = Operating System Trust	
3.2 = Trust Between Computer Systems	
3.3 = Border Guard Trust	
2. Some Trust Statements are restated for the Communications Trust category since new Trust Solutions apply (again, see Appendix A for details).

## 5. SUMMARY

The two illustrative examples, the web-server and the remotely administered system, introduced in Section 2, provided an introduction to the concept of a trust model. Section 3 followed with a detailed examination of the fundamentals of a trust model. It described trust relationships at various levels within a computing system and for various configurations, beginning with the trust inherent in modern computer system,

and building to the trust relationships imposed on highly interconnected and interdependent systems. Section 4 built on the trust model concepts by applying the fundamentals covered in Section 3 to the two illustrations introduced in Section 2.

Tables 4 and 5 contain the trust statements that constitute the two trust models developed. Table 4, with its trust statements for the Web-Server Trust Model, provides the context for the operation of the web-server, the other systems connected to the web-server, and the network on which the web-server resides. While the web-server provides access to the public, its trust of the public and those who access the computer for other reasons can now be limited by applying the trust model. Likewise, Table 5 provides the context for the operation of the remotely administered system, the other systems connected to it, and the network to which it is connected. While the remotely administered system provides access to valid users, its trust of these users and even the remote administrator can now be limited.

Trust models allow a system administrator to visualize the impact of users and the system configuration in terms of trusts. These trusts can be grouped (e.g., access, data, operations, and connectivity trusts) to aid in understanding and ensure completeness. Trust models allow risks to be identified and mitigated. For example, the risk associated with allowing public access to the web-server was mitigated by reducing trust in the operating system, trust in the application, and trust in the network connectivity, while implementing rules with Border Guards. As an additional benefit, this layered approach to security will help prevent situations in which a single point of failure allows the system to be compromised.

A well-developed trust model can lend itself to reuse. The Web-Server Trust Model, once developed, can now serve a baseline for trust models of similar systems. As illustrated, the Remotely Administered System Trust Model was derived, with only slight modification and the addition of two trust statements, from the Web-Server Trust Model. Trust models are not, however, cast in stone. They should be viewed as living models. As conditions change, the trust statements (and the trust solutions) may need to be weakened or strengthened. Periodic reviews and revisions are essential to keep the trust models current with developing technologies, threats, and vulnerabilities. One should keep in mind that a trust model that describes who is let-in, rather than who is kept out, is more likely to expose unintentional security vulnerabilities and weaknesses, and thereby enhance risk mitigation.

In conclusion, a trust model is a tool that aids in analyzing and enhancing a system's security posture. It does so by establishing an understanding and creating a view of the trust relationships shared by individuals, computer systems, and networks, along with the associated risks that are inherent with granting this trust. Implementing trust models can shape a computing environment based on varying degrees of trust where merited, and minimal trust where not merited.



**COMPLETE DEVELOPMENT OF A WEB-SERVER TRUST MODEL**

Table A - Detailed background information (i.e., trust statement, trust solution, and justification/discussion) used for building the Web-Server Trust Model

Trust Statement	Trust Solution	Justification/Discussion
<b>1.0 Access Trust</b>		
1.1 The web-server provides access to information for the public.	Use a dedicated machine for the web-server. Restrict/limit the operating system services, other applications, and connections to other machines.	It is important that the web-server not run other server software or services, such as domain name server (DNS), email, and simple network management protocol (SNMP). Since the web-server is publicly known and accessible, it provides an attractive target for those trying to exploit a system. Running superfluous software provides additional avenues of attack and increases the impact of a compromise. If the web-server is compromised, and that same computer is also running DNS and/or SNMP, the entire site could be at risk.
1.2 The web-server operating system and web applications limit access for actions such as updating web pages to only administrators and specific users with a need.	No regular users have accounts on the web-server and no user's account should be shared with other machines.	Reducing the number of users with write access to the web-server lessens the chance of accidental and malicious change. It limits the possible damage to other systems if user accounts are not shared with other systems at the local site.
1.3 The web-server operating system limits non-web access to only authorized users.	Use TCP-wrappers, SSH, or IPSEC to ensure that only authorized users can access the web-server computer using non-web means.	TCP-wrappers can limit access based on IP address. Therefore, Telnet access could be limited only to those coming from allowed IP addresses. TCP-wrappers also provide logging of allowed and denied accesses. SSH and IPSEC require a shared password between systems and provide additional security through encryption.
1.4 The web-server application program grants only selected users administrative access.	Restrict administrative permission to only IP addresses of those with permission to perform the administration.	To reduce the vulnerability incurred by allowing multiple avenues of access, the configuration should allow administrative privilege to only certain IP addresses.
1.5 The web-server application program grants access to sensitive data on the web-server to only selected users.	Control access to sensitive data on the web-server with user IDs and passwords. If the users use only certain computers, additional restrictions can be applied to require that they come from one of the approved IP addresses.	Access to sensitive data that should not be publicly available should require a user ID and password. Obfuscation should not be used as a means of protection. Additional protection of the user ID and password should be implemented with a secure web-server. IP address restrictions provide additional protection so that if a user ID and password are compromised the attacking machine must be within the allowed IP address range.
<b>2.0 Data Trust</b>		
2.1 The web-server may not directly access the file system on other systems.	No file system from other machines is accessible (i.e. mapped for Windows and mounted for Unix).	Making file systems available/accessible provides possible paths to corrupt data on the other system. The convenience of being able to quickly transfer data between the web-server and other system does not outweigh the risks incurred.
2.2 Other systems may not directly access the file system on the web-server.	No file system on the web-server is accessible from other systems.	
2.3 Web-server data access is limited to only those users that need access.	Restrict world and group permissions for access to data.	Giving permission to access and change or update data to only those that provided the data, limits the risk of accidental and malicious data corruption.
2.4 The web-server application program protects web-server data from accidental release.	Configure the web-server application program so that directory listings are not displayed.	Allowing the display of the directory listing can make the files and subdirectories available to the web user. This may expose outdated, incomplete, or unapproved data to download. Since the directories are also visible, the directory structure also can be examined and exploited.



Table A - Detailed background information (i.e., trust statement, trust solution, and justification/discussion) used for building the Web-Server Trust Model

Trust Statement	Trust Solution	Justification/Discussion
2.5 The web-server application program protects database data from change, deletion, and compromise.	<p>Configure the web-server application program so that the database files, either on the web-server or on a separate database server, cannot be corrupted.</p> <p>If the database server is running on the web-server, ensure that access to the data is only via the database server. Restricting the file and directory permissions on the database server can do this. Also, limit database access permission to only the web-server administrator and selected users.</p> <p>If another machine is used for the database server, it should be protected from attacks from the web-server. Restrict communications between the servers via IP address and port using a router or TCP-wrappers whenever possible. This will limit the type of attacks that a compromised web-server could use and prevent scans from other sources. SSH and IPSEC can provide additional protection by providing authentication and encryption between the servers.</p>	Any database server that allows a publicly accessible web-server to request and update its data is a target for attacks. These attacks can change, delete, or corrupt data that may be used by other systems. The attacks can be direct attacks against the database or indirect attacks by using the web-server.
<b>3.0 Operations Trust</b>		
3.1 The web-server operating system functions in a well-known and proper manner.	Use the most current version of operating system and keep the system updated with current patches.	In the past, it was common to use an outdated computer with an old operating system as the dedicated web-server. This is no longer a smart thing to do. Because of the number of attacks the web-server is subjected to, the web-server should be run on a computer with a current supported version of the operating system. As exploits are identified, vendors will release patches to stop them. The installation of updates is a continuous process. Since the tools needed to activate the exploit are frequently available on websites, failure to maintain an updated operating system accounts for a large number of compromised web-servers. [21, 22]
3.2 The web-server operating system minimizes the exposure from unused services.	Activate only those services that are needed and deactivate those services that are not needed.	Default computer configurations enable a large number of services that make the computer vulnerable to attack. These services may be used as points-of-entry for internal attacks and could enable denial of service attacks (e.g., the use of chargen and echo). The services that should be disabled include: echo, chargen, TFTP, DNS, finger, POP2, POP3, Sunrpc, nntp, imap, who, printer, and NFS. Others, if not needed, should also be disabled, such as: systat, Telnet, FTP, login, cmd, and date. Disabling the SMTP daemon will prohibit incoming email to or via the web-server, which is another justification for a dedicated machine. Email can be explicitly invoked if mail needs to be sent.
3.3 The web-server operating system limits denial of service attacks.		
3.4 The web-server operating system has installed only those programs needed for web-server operation.	Do not install programs that are not needed, especially those that would aid an attacker if the computer is compromised, such as compilers, snooping programs, etc.	If the system is compromised, it can be used to attack other systems on the network. Limiting the tools available to a hacker, forcing him/her to download and install the software needed to attack other systems, may at least buy the time necessary to detect and stop his/her actions.

Table A - Detailed background information (i.e., trust statement, trust solution, and justification/discussion) used for building the Web-Server Trust Model

Trust Statement	Trust Solution	Justification/Discussion
3.5 The web-server operating system's integrity is protected from unauthorized changes.	Restrict and minimize world and group permissions for the operating system software. Create, and store in a secure location, an operating system benchmark or "fingerprint." Periodically, and especially if/when the web-server has been attacked, the operating system fingerprint in use should be compared with the original benchmark fingerprint to assess and recover from possible damage.	Some operating systems have very open permissions that allow a normal user to overwrite system commands. This allows Trojan programs to be executed, conceivably by unsuspecting users. Restricting the permissions limits a user to only the access required.
3.6 The web-server application program is protected from known malicious sites.	Block access (in the web-server configuration files) to IP addresses of untrusted sites.	The IP addresses of those sites that may attack the web-server or the site network should be blocked as a precaution against attack and compromise. This block should also include sites that probe the web-server or site network on a regular basis.
3.7 The web-server application program functions in a well-known and proper manner.	Use current versions of application programs and keep them updated with current patches.	Attackers constantly scan web-server application programs for weaknesses. As new vulnerabilities and exploits are identified, vendors attempt to release patches as soon as possible. The installation of updates is a continuous process. Since the tools needed to activate the exploits are frequently available on websites, failure to maintain an updated web-server application program can lead to the compromise of sensitive data and/or public embarrassment.
3.8 The web-server application helper programs function in a well-known and proper manner.	Test Java programs, CGI and Perl scripts, and other executable programs for security holes. This includes validation of actions when malicious data is sent back to the web-server (e.g., running Perl with the Taint option enabled). Remove old, untested, and unused code from operational directories.	Web-server helper programs are frequently the targets of attacks. Malicious users try to break poorly written code or code that does not provide protection against malicious data. When these exploits are found, the system is compromised since the malicious user can use these programs as a means to run arbitrary commands on the web-server.

#### 4.0 Communications Trust

Trust Statement (restated)	New Trust Solution	Justification/Discussion
1.2 The web-server operating system and web applications limit access for actions such as updating web pages to only administrators and specific users with a need.	<p>In addition to the other actions previously described in terms of Access Trust for Trust Statement 1.2, Configurations 2 and 4 (reference Table 3) provide minimal trust of unauthorized computers attempting to access the operating system and web application or attempting to change web pages. Restrict, via access control lists or filters, non-web access (e.g., Telnet, FTP, etc.) to only certain IP addresses. An entry would be in the form of:</p> <p>(From approved IP address AND To web-server IP address AND Telnet): Permit</p> <p>This rule does not protect against malicious users coming from approved IP addresses, but it does minimize the trust between the web-server and other computers for non-web access. Configuration 3 could provide protection against non-web access from the external network, but it provides no protection against the site internal network.</p>	Explicitly allowing non-web access from only selected computers denies access from all other computers. Therefore, the possibility of non-web attack is limited to the selected (hopefully trusted) computers.

<b>4.0 Communications Trust</b>		
<b>Trust Statement (restated)</b>	<b>New Trust Solution</b>	<b>Justification/Discussion</b>
<p>2.1 The web-server may not directly access the file system on other systems.</p> <p>2.2 Other systems may not directly access the file system on the web-server.</p>	<p>In addition to the other actions previously described, Configurations 2 and 4 provide minimal trust of computers attempting to directly access the web-server file systems and of the web-server attempting to directly access other computer file systems. Block, via access control lists or filters, the ports used by mapping and mounting protocols. No rule would be required because the last rule automatically blocks this type of access.</p>	<p>Reducing trust to deny all mapping or mounting protocols prevents the sharing of file systems. If the web-server is compromised, it would also prevent the attacker from mapping or mounting other computer file systems including those not part of the site internal network. Configuration 1 would not prevent computers on the external network from direct access to the web-server file system, nor would it prevent the web-server from direct access to the computer file systems on the external network. The limitations of Configuration 3 are similar to those of Configuration 1, except that the threat comes from computers on the site internal network.</p>
<b>Trust Statement</b>	<b>Trust Solution</b>	<b>Justification/Discussion</b>
<p>4.1 The web-server will limit damage if compromised.</p>	<p>Apply Configuration 2 or 4 to provide minimal trust to external and internal connections.</p>	<p>All four configurations provide public access to information. Configuration 1 provides the least protection since the web-server is protected only by the computer and application configuration. Configuration 3 provides protection against external network attacks, but does not provide protection against site internal network attacks, nor does it protect the site if the web-server is compromised.</p>
<p>4.2 Communication is limited to only that which is necessary with other local site systems.</p>	<p>Minimize trust of non-web network access (e.g., Telnet, FTP, rlogin, etc.) by limiting non-web access to only to those IP addresses and ports that are required. Application of Configuration 4 is best since only it can accomplish this for both external and internal access.</p>	<p>Telnetting and FTPing to/from the web-server should not be enabled for the public. Limit Telnet and FTP access to only the IP addresses trusted.</p>
<p>4.3 The Border Guards minimize the exposure from unused services.</p> <p>4.4 The Border Guards limit denial of service attacks.</p>	<p>Apply Configuration 2 or 4 to block those services not needed by the web-server. No rule would be required because the last rule automatically blocks this type of access.</p>	<p>In addition to the solutions applied with Trust Statements 3.2 and 3.3, Border Guards (via Configurations 2 and 4) can provide an additional layer of protection by placing minimal trust on the questionable/un-needed services by blocking them.</p> <p>Configuration 1 could protect from services that are not needed from the site internal network, but it provides no protection against the external network. Configuration 3 could protect from services that are not needed from the external network, but provides no protection against the site internal network.</p>
<p>4.5 The Border Guards protect the web-server from known malicious sites.</p>	<p>Apply Configuration 2, 3, or 4 to block malicious sites. Restrict, via access control lists or filters, all access to certain IP addresses. An entry would be in the form of:</p> <p>(From disapproved IP address AND To web-server IP address AND any protocol): Deny</p> <p>This rule protects the web-server against malicious users only when they come from an IP address that is blocked. It does not provide protection if the malicious user uses an approved IP address that is not blocked. To also protect the site internal network, use:</p> <p>(From disapproved IP address AND To any host AND any protocol): Deny</p> <p>Configuration 1 would provide no protection against malicious sites.</p>	<p>The IP address of sites that attack the web-server or the site network should be blocked as a precautionary measure. This could also include sites that probe the web-server or site network on a regular basis and those that attempt denial of service attacks.</p>

**REFERENCES**

1. *Practical UNIX & Internet Security*, Garfinkel, S. and Spafford, G., O'Reilly & Associates, Inc., 1996
2. *Solaris Security Step By Step*, V1.0, SANS Institute, 1999, <http://www.sans.org/>
3. *Windows NT Security Step By Step*, V1.0, SANS Institute, 1999
4. *Linux Security Step By Step*, V1.0, SANS Institute, 1999
5. *Firewalls and Internet Security: Repelling the Wily Hacker*, Cheswick, W. and Bellovin, S., Addison-Wesley, 1994
6. *TCP-Wrappers*. Free Unix network logger software by Wietse Venema, also known as TCPD or LOG\_TCP. The program logs the client host name of incoming Telnet, FTP, rsh, rlogin, and finger requests. Security options are: access control per host, domain, and/or service; detection of host name spoofing or host address spoofing; booby traps to implement an early-warning system. <ftp://ftp.porcupine.org/pub/security/index.html>.
7. *BlackIce Defender*. BlackIce Defender is commercial software from Network ICE that will detect intruder attempts, identify who they are, and stop them. Running on any Windows 95, 98, or NT based system, BlackIce Defender consists of a sophisticated network monitoring engine that can scan all inbound and outbound traffic on a PC for suspicious activity. On finding an attempt to breach the computer, BlackIce denies the hacker access to the computer while leaving the legitimate traffic unaffected. <http://www.networkice.com/Products/BlackICE/default.htm>
8. *BackOfficer Friendly*. BackOfficer Friendly is free software from Network Flight Recorder, Inc. that provides a useful little burglar alarm which is simple, unobtrusive, and easy to install. It identifies attacks from BackOrifice, as well as other types of scans. Windows and Unix versions are available. <http://www.nfr.net/products/bof/>
9. *Nuke Nabber*. NukeNabber is free software for Windows 95, 98, and NT systems. It listens on TCP and UDP ports commonly attacked over the Internet. A total of 50 ports can be monitored simultaneously. <http://www.dynamsol.com/puppet/nukenabber.html>
10. *ZoneAlarm*. ZoneAlarm is commercial software for Windows 95, 98, and NT systems that provides comprehensive protection. ZoneAlarm incorporates a firewall, an application control, and an Internet lock, dynamically assigned to Security Levels and Zones. <http://www.zonelabs.com/>

11. *Internet Firewalls and Security*. This 3COM technical paper discusses how an Internet firewall fits into an organization's overall security policy and describes several types of firewall systems and their advantages and disadvantages. [http://www.3com.com/technology/tech\\_net/white\\_papers/500619s.html](http://www.3com.com/technology/tech_net/white_papers/500619s.html)
12. *Healthcare Information Security: How a Secure Data Network Can Help Healthcare Organizations Meet the Challenge of Healthcare Security Regulations*. This 3Com white paper focuses on the stringent information security policies and rules established by HIPAA. [http://www.3com.com/technology/tech\\_net/white\\_papers/503069.html](http://www.3com.com/technology/tech_net/white_papers/503069.html)
13. *CheckPoint FireWall-1® Technical Overview*. This document describes many of the features of CheckPoint FireWall-1. A step-by-step procedure demonstrates how to build a FireWall-1 Rule Base and install a Security Policy for a simple network configuration. <http://www.checkpoint.com/products/downloads/fw1-4.0tech.pdf>
14. *CERT® Security Improvement Modules*. Each CERT Security Improvement module addresses an important but narrowly defined problem in network security. It provides guidance to help organizations improve the security of their networked computer systems. Each module page links to a series of practices and implementations. Practices describe the choices and issues that must be addressed to solve a network security problem. Implementations describe tasks that implement recommendations described in the practices. <http://www.cert.org/security-improvement/>
15. *CERT® Technical Tips*. Tech tips provide basic information on a variety of Internet security issues. [http://www.cert.org/tech\\_tips/](http://www.cert.org/tech_tips/)
16. *CERT® Deploying Firewalls*. This is one of the Security Improvement Modules that can be found in reference [14]. The practices in this module will address designing, installing, and deploying firewalls. <http://www.cert.org/security-improvement/modules/m08.html>
17. *TCPDump*. TCPDump is a Unix packet capture program written by Steve McCanne, Craig Leres, and Van Jacobson of Lawrence Berkeley Lab. SUN Solaris machines come with a version of TCPDump called snoop. <ftp://ee.lbl.gov/tcpdump.tar.Z>
18. *Remote OS detection via TCP/IP Stack FingerPrinting*, <http://www.insecure.org/nmap/nmap-fingerprinting-article.html>
19. CERT® Coordination Center, <http://www.cert.org/advisories/>
20. National Infrastructure Protection Center <http://www.fbi.gov/nipc/welcome.htm>

21. *PC Week* series on hacking its website.


- *Hack this: PC Week Labs site begs attack* - September 20, 1999  
<http://www.zdnet.com/pcweek/stories/news/0,4153,2336675,00.html>
- *Attacked and hacked!* - October 11, 1999  
<http://www.zdnet.com/pcweek/stories/news/0,4153,2350743,00.html>
- *The Gibraltar hack: Anatomy of a break-in* - October 11, 1999  
<http://www.zdnet.com/pcweek/stories/jumps/0,4270,2350744,00.html>

22. "Hacker attack latest in string of online credit card thefts," John Borland, *CNET News.com*, March 2, 2000. Thousands of credit cards numbers were stolen from an e-commerce site through a security hole for which a patch has been available for a year and a half. <http://news.cnet.com/category/0-1007-200-1563391.html>

23. *Commonly Probed Ports*, Toby Miller, SANS Institute - Global Incident Analysis Center. This web page provides a table of ports that are commonly probed with the known Trojans or applications that run on these ports.  
<http://www.sans.org/y2k/ports.htm>

24. *Sudo*: Sudo (superuser do) allows a system administrator to give certain users (or groups of users) the ability to run some (or all) commands as root or another user while logging the commands and arguments. Sudo is free software and is distributed under a BSD-style license. <http://www.courtesan.com/sudo/sudo.html>

25. *DHIAP Phase I Technology Demonstration Report: Prototype for Remote Authentication Dial-In User Service (RADIUS)*, Advanced Technology Institute Information Protection Solutions Report 00-04, Contract Number DAMD17-99-C-9001, April 2000.




\* EXCERPTS FROM \*

**Defense Healthcare Information  
Assurance Program  
Business Case Analysis**

May 15, 2000

Defense Healthcare Information Assurance Program

Slide 1




**Meeting Purpose and Approach**

- Introduce the Draft BCA Methodology
- Review scope and approach for BCA #1, RADIUS
- Introduce candidate topics for BCAs 2-4

BCA Project

Defense Healthcare Information Assurance Program

Slide 2




**BCA Methodology - Major Activities**

- Establish the Scope of the BCA
- Collect Background Information
- Develop BCA Plan and Schedule
- Develop & Refine Evaluation Criteria
- Collect Data
- Analyze Data
- Develop Conclusions
- Report
- Refine Methodology

Business Case Methodology

Defense Healthcare Information Assurance Program

Slide 3




**Establish the Scope of the BCA**

- Exactly what technology or process are we analyzing?
- Does this apply
  - To all services?
  - To all size MTFs?
  - Fixed (CONUS-OCONUS), mobile, shipboard?
  - To Medical Commands rather than MTFs?
- Establish technological bounds

Business Case Methodology

Defense Healthcare Information Assurance Program

Slide 4




**Collect Background Information**

- What is this intended to do?
- Does it work?
  - Is it in use anywhere?
  - What does open literature say?
- Review vendor claims & literature
- Review independent research reports

Business Case Methodology

Defense Healthcare Information Assurance Program

Slide 5



**Develop Plan & Schedule**


- Tailor standard plan to specific BCA
  - Identify participating MTFs and/or parent commands
  - Any unique elements
    - tasks?
    - schedule impacts?
    - cost drivers?
- Assign tasks
- Estimate time lines

Business Case Methodology

Defense Healthcare Information Assurance Program

Slide 6




**Refine Evaluation Criteria** 

- Cost Factors
- Non-cost Factors
- Risk Factors
- Economic Analysis

Business Case Methodology Defense Healthcare Information Assurance Program


Slide 7

**Evaluation Criteria - Cost** 

- Conversion / Implementation
  - Requirements analysis
  - Design
  - Development
  - HW & SW purchase
  - Site preparation
  - System configuration
  - Initial training

Business Case Methodology Defense Healthcare Information Assurance Program


Slide 8

**Evaluation Criteria - Cost (cont'd)** 

- Sustainment (Pre-, During, and Post)
  - Personnel
    - ◆ Salaries
    - ◆ Training
  - Supplies
  - Energy
  - Maintenance
  - Space
  - Administrative cost to contract
  - Contract cost

Business Case Methodology Defense Healthcare Information Assurance Program


Slide 9

**Evaluation Criteria - Non-Cost** 

- Functionality
- Compliance with regulations
- User acceptance
  - Quality of service
  - Utility
  - Ease of use
- Reliability, availability, maintainability
- Expandability, flexibility, scalability

Business Case Methodology Defense Healthcare Information Assurance Program


Slide 10

**Evaluation Criteria - Risk** 

- Financial – unanticipated costs?
  - Conversion / implementation
  - Post conversion sustainment
- Technical – do what it's supposed to do?
  - Untested functions
  - New environment
- Schedule – meet conversion milestones?
  - What is the impact?
  - What is the critical path?

Business Case Methodology Defense Healthcare Information Assurance Program

Slide 11

**Evaluation Criteria - Economic Analysis** 

- Cost Benefit Ratio
- Net Present Value
- Present Value

Business Case Methodology Defense Healthcare Information Assurance Program

Slide 12



### Data Collection

- Determine data sources
  - Individuals using technology
  - Information systems
- Identify collection tools
  - Usage logs
  - User satisfaction surveys
- Collect data
  - In-person interviews
  - Analysis of logs and surveys

Business Case Methodology Defense Healthcare Information Assurance Program

Slide 13

### Analyze Data (Example)

- Substantial cost to implement
- Some cost (# FTE) to sustain
- Non-compliant before implementation
- Low risk of implementation

Phase	Cost	Non-Cost	Risk
Pre	~480	~100	~100
Mid	~100	~480	~100
Post	~100	~100	~150

Business Case Methodology Defense Healthcare Information Assurance Program

Slide 14

### Develop Conclusions / Recommendations (Example)

- Compliance is mandatory
  - Likely to fail JCAHO
  - Risk legal penalties
- Implementation improves (does not perfect) compliance
- Recommendation: Implement, but centralize sustainment
- Recommendation: Migrate to all MEDCOM

Business Case Methodology Defense Healthcare Information Assurance Program

Slide 15

### Report

- RIMR format and / or paper
- Graphics

Business Case Methodology Defense Healthcare Information Assurance Program

Slide 16

### Refine Methodology

- Criteria
- Collection Methods
  - Tools
  - Audience
  - Depth
- Analysis Methods
- Align with Existing Methods
- Reporting Formats

Business Case Methodology Defense Healthcare Information Assurance Program

Slide 17

### RADIUS BCA - Scope

Evaluation of RADIUS prototype now in use at DDEAMC and WACH

- Pre- and Post-Conversion Analysis that considers
  - Cost factors of WACH and DDEAMC sites
  - Non-cost factors at WACH and DDEAMC
    - Information Management
    - User
  - Risk Factors
  - Economic analysis
- Conversion Analysis that includes
  - Costs
  - Effort

BCA #1: RADIUS Defense Healthcare Information Assurance Program

Slide 18

**RADIUS BCA - Approach**

- Adapt BCA Methodology to the RADIUS topic and scope
- Where appropriate, use information existing in files at ATI, LMES, WACH, DDEAMC
- Collect remaining data using surveys, VTCs, and (if necessary) meetings at sites
- Perform analysis and develop conclusions
- Review draft report with sites
- Publish final report

BCA #1: RADIUS

Defense Healthcare Information Assurance Program

Slide 19


**BCA Topics****BCA # 1 – RADIUS - “as is”****BCA #2 - 4**

- Biometrics (authentication)
- Role-Based Access (insider threat)
- Digital Watermarking (integrity)
- Intrusion Detection (external threat)
- Virus Mitigation (prevention & recovery)
- Firewalls (external threat)
- Dictation encryption (dictation/transcription protection)
- Others

Topics for  
BCAs #2-4

Defense Healthcare Information Assurance Program

Slide 20



---

\* EXCERPTS FROM \*


**Defense Healthcare Information  
Assurance Program  
Business Case Analysis**

June 1, 2000

---

Defense Healthcare Information Assurance Program

Slide 1



---

**BCA 2-4 Topic Selection Process**


- Derive Candidate Topics
- Screen Initial Candidates
- Evaluate/Prioritize Candidates
- Select

---

Topics for  
BCAs #2-4

Defense Healthcare Information Assurance Program

Slide 2



---

**Derived Candidates**


- Role-based access
- User friendly authenticated
- Network audit ability
- Virus mitigation
- Intrusion detection
- Dictation protection
- User profiling
- Computer-based security education
- Personnel security measures
- Rapid authentication/rapid log-off
- Public key infrastructure
- Security policy review and amendment

---

Topics for  
BCAs #2-4

Defense Healthcare Information Assurance Program

Slide 3



---

**Screened Candidate List**


- Computer-based security education
- Dictation protection
- Network auditability
- Rapid authentication/rapid log-off
- Role-based access control
- User-friendly authentication
- User profiling

---

Topics for  
BCAs #2-4

Defense Healthcare Information Assurance Program

Slide 4



---

**Evaluate / Prioritize Candidates**

- Customer priorities
- Team Expertise
- Potential Benefit
- Compliance
- Impact
- Tri-service
- Risks
- Breadth of applicability
- Existing program


sum (weights x raw scores) = overall score

---

Topics for  
BCAs #2-4

Defense Healthcare Information Assurance Program

Slide 5



---

**Prioritized Topics**

■ Role-based access control	- 515
■ Network auditability	- 507
■ User-friendly authentication	- 488
■ Rapid authentication/rapid log-off	- 410
■ User profiling	- 376
■ Computer-based security education	- 377
■ Dictation protection	- 352

---

Topics for  
BCAs #2-4

Defense Healthcare Information Assurance Program

Slide 6

**Role-Based Access Control**

- Access based on role, not just identity
  - Physician, administrator, lab tech, insurer
  - Oversight of patients (floor, names, type treatment)

## ■ Pros

- Supports compliance with HIPAA
- Supports data integrity
- Any solution will be costly – makes decision critical

## ■ Cons

- No known easy solution for MTF environment

BCA 2-4

Defense Healthcare Information Assurance Program

Slide 7

**User-Friendly Authentication**

- Effective and suited to all environments
  - Perhaps choice of biometrics
  - Effective password creation & management
  - Tokens
  - Combinations

## ■ Pros

- Encourages individual log-on
- Identifies multiple users of single workstation

## ■ Cons

- Large number of authentication methods

BCA 2-4

Defense Healthcare Information Assurance Program

Slide 8

**Network Auditability**

- Audit transactions across network
  - Who logged, from where?
  - What system was accessed, what application?
  - What patient info was seen, changed, transferred?

## ■ Pros

- Enables compliance with HIPAA
- Deters insider threat

## ■ Cons

- Little information available on products

BCA 2-4

Defense Healthcare Information Assurance Program

Slide 9

**Common Attributes of Top Three**

- Support HIPAA Information Security
- Tend to Form an Integrated Set
- Address the Insider Threat
- Broad Topics

BCA 2-4

Defense Healthcare Information Assurance Program

Slide 10


**Broad BCA Topics**

- These are not technologies or products awaiting analysis.
- They are general solutions to major problems.
- We need to research these and scope down to implementable technologies.
- We will review them with you again.

BCA 2-4

Defense Healthcare Information Assurance Program

Slide 11




## Defense Healthcare Information Assurance Program

An opportunity to participate in an information security risk analysis program, led by experts and funded by the US Army Medical Research and Materiel Command

*Defense Healthcare Information Assurance Program*

Slide 1




## Information Security Risk Analysis

Facilitated workshops with multiple organizational levels

- Begins with Senior Management
  - Set scope of evaluation
  - Identify information assets and perceived threats
  - Identify current strategic health information assurance activities
- Works with Operational Area Management
  - Identify information assets and perceived threats from their perspectives
  - Identify current operational approach to health information assurance
  - Select staff to participate
- Works with Staff Members
  - Identify information assets, perceived threats, and current operational realities from their perspectives
  - Outline appropriate mitigation practices

*Defense Healthcare Information Assurance Program*

Slide 3




## Requirements

- Endorse concept
- Briefing with senior staff from DHIAP team
- Dedicate resources
  - Senior management (1/2 day)
  - Clinical, administrative, and IT staff members (1/2 day)
  - Multi-disciplinary Analysis Team (2 1/2 days)
- Participate in Senior Management workshop
- Support staff participation in Operational Area Manager workshops, Staff workshops, and Risk Analysis Planning
- Designate operational areas of concentration

*Defense Healthcare Information Assurance Program*

Slide 5



## Context

- Defense Healthcare Information Assurance Program (DHIAP)
  - Designed to identify risks and vulnerabilities in military medical information systems and organizational practices
  - Building tool sets to aid military Medical Treatment Facilities in prioritizing and addressing the risks and vulnerabilities
- Voluntary program requiring commitment from Commander and supporting management levels

*Defense Healthcare Information Assurance Program*

Slide 2

## Wins for MTF

- Risk awareness and planning introduced into organization
- Expert technical assistance at no cost (\$) to site
- Preparation for JCAHO
- Development of internal capability to identify vulnerabilities, risks, threats, mitigation approaches
- Consequence avoidance
  - Understand HIPAA (information security) regulations and potential impact
  - Proactive approach to dealing with HIPAA requirements

## Wins for Department of Defense

- Build and validate health information assurance risk analysis tools and techniques
- Potential to investigate inter-organizational risk management issues
- Identify health information assurance risk issues common to multiple sites
- Contribute to repository of risks to military healthcare information

Slide 4

## Results

- Identification and prioritization of risks-assets, threats, vulnerabilities
- Protection strategy for high priority risks
- Plan for focus on next-tier risks

## Benefits

- Understanding of current information security risks
- Build organizational awareness and capability in information security risk management

## Investment Required

- Command sponsorship and participation
- Commitment of personnel resources

## Workshops

- Risk Analysis discussions, facilitated by experts
- Expert-led technical vulnerability assessment
- Expert-led planning for mitigation efforts

Slide 6





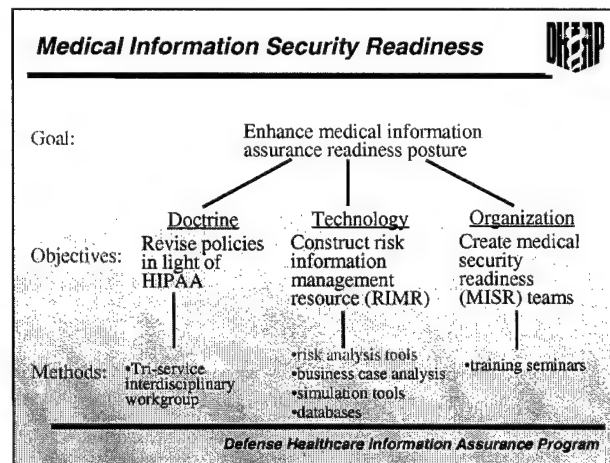
Slide 1

**Overview**

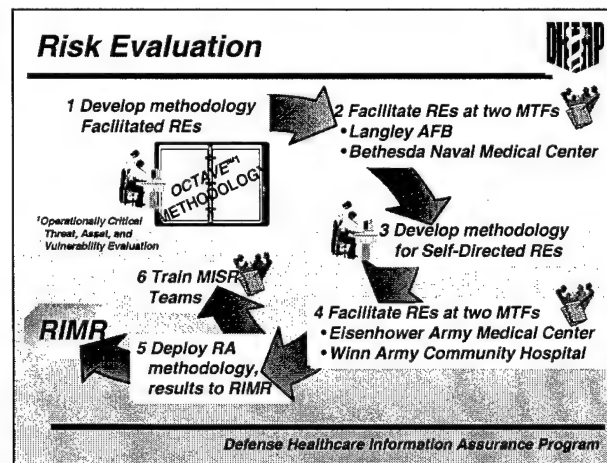
- **Originated as Congressional mandate to**
  - Evaluate DoD healthcare systems' protection of patient information
  - Demonstrate system improvements
- **Awarded and managed by Telemedicine and Advanced Technology Research Center (TATRC)**
- **Goals**
  - Develop and apply tools and techniques that
    - Evaluate system and network risks
    - Address necessary doctrine, infrastructure, training, programmatic, and technology issues
    - Support Medical Information Security Readiness (MISR) strategy
    - Support Risk Information Management Resource (RIMR)
  - Improve information assurance for Army Medical Treatment Facilities

Defense Healthcare Information Assurance Program

Slide 2



Slide 3



Slide 4

**Enrollment of Sites**

**Pilots for the RE methodology**

- Two sites for REs facilitated by the DHIAP Team
- Two sites for self-directed REs mentored by the DHIAP Team

**Site Characteristics**

- Representative of larger activities
- Willing to commit to effort required (meetings, working groups, collaborative tasks)
- Willing to leverage results for benefit of site and service
- Willing to share findings with TATRC for RIMR

Defense Healthcare Information Assurance Program

Slide 5

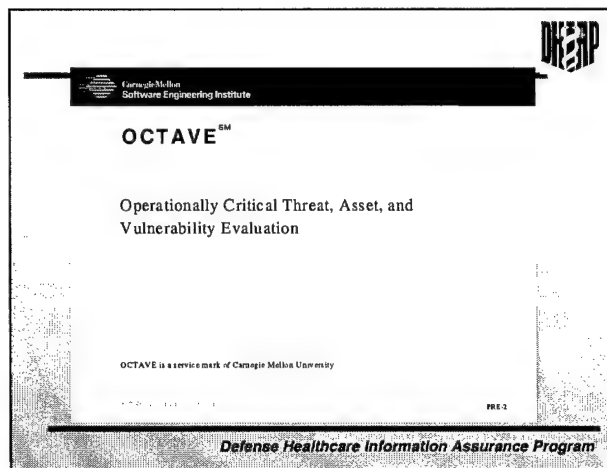
**Facilitated Information Security Risk Evaluation**

Software Engineering Institute  
Carnegie Mellon University  
Pittsburgh, PA 15213

Sponsored by the U.S. Department of Defense

Defense Healthcare Information Assurance Program

Slide 6



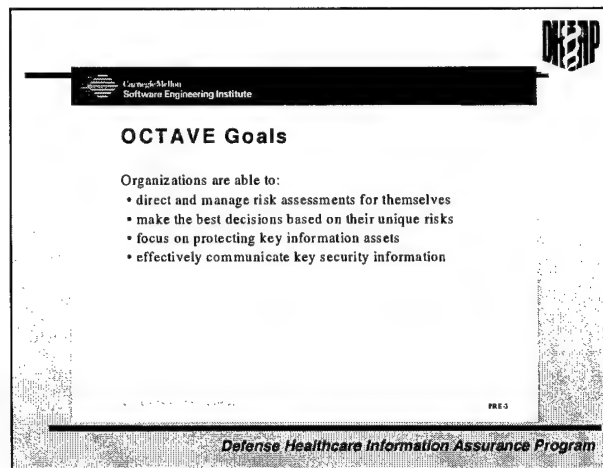
**OCTAVE<sup>SM</sup>**

Operationally Critical Threat, Asset, and Vulnerability Evaluation

OCTAVE is a service mark of Carnegie Mellon University

Defense Healthcare Information Assurance Program

Slide 7



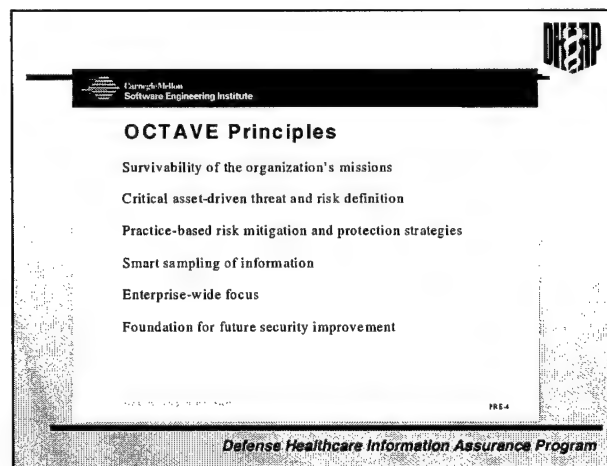
**OCTAVE Goals**

Organizations are able to:

- direct and manage risk assessments for themselves
- make the best decisions based on their unique risks
- focus on protecting key information assets
- effectively communicate key security information

Defense Healthcare Information Assurance Program

Slide 8

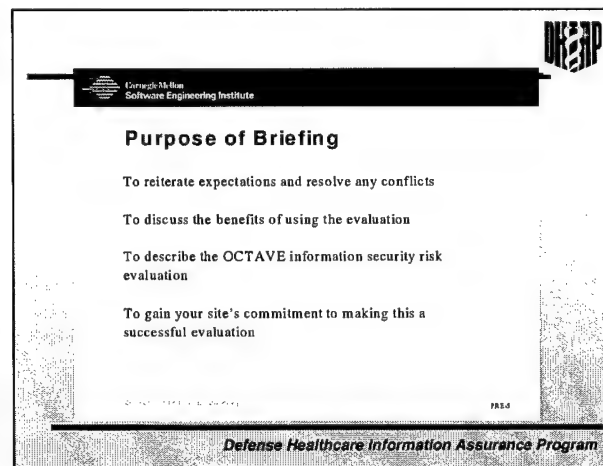


**OCTAVE Principles**

- Survivability of the organization's missions
- Critical asset-driven threat and risk definition
- Practice-based risk mitigation and protection strategies
- Smart sampling of information
- Enterprise-wide focus
- Foundation for future security improvement

Defense Healthcare Information Assurance Program

Slide 9

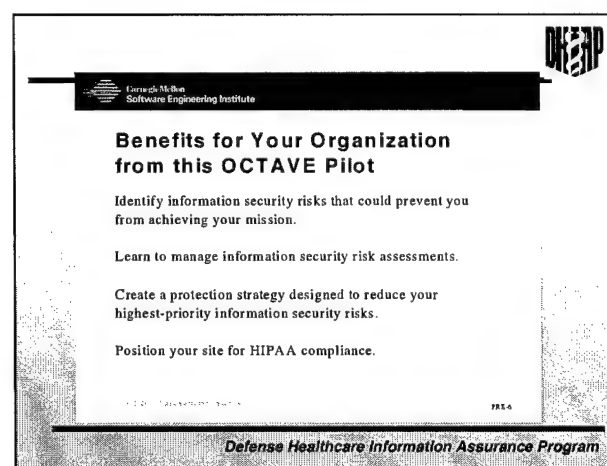


**Purpose of Briefing**

- To reiterate expectations and resolve any conflicts
- To discuss the benefits of using the evaluation
- To describe the OCTAVE information security risk evaluation
- To gain your site's commitment to making this a successful evaluation

Defense Healthcare Information Assurance Program

Slide 10

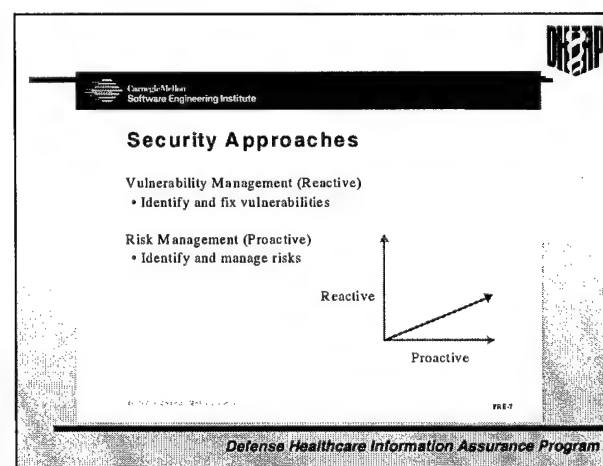


**Benefits for Your Organization from this OCTAVE Pilot**

- Identify information security risks that could prevent you from achieving your mission.
- Learn to manage information security risk assessments.
- Create a protection strategy designed to reduce your highest-priority information security risks.
- Position your site for HIPAA compliance.

Defense Healthcare Information Assurance Program

Slide 11



**Security Approaches**

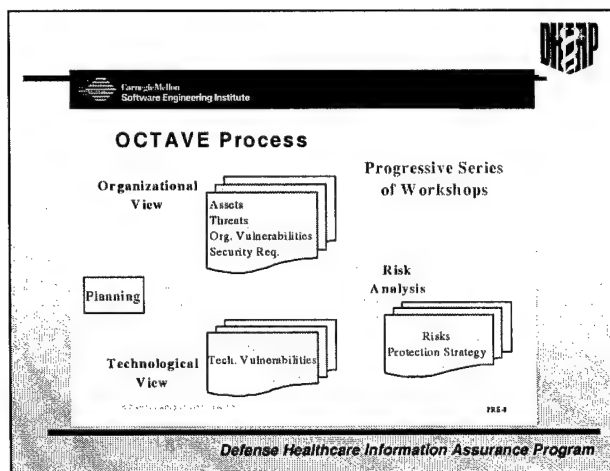
- Vulnerability Management (Reactive)
  - Identify and fix vulnerabilities
- Risk Management (Proactive)
  - Identify and manage risks

Reactive

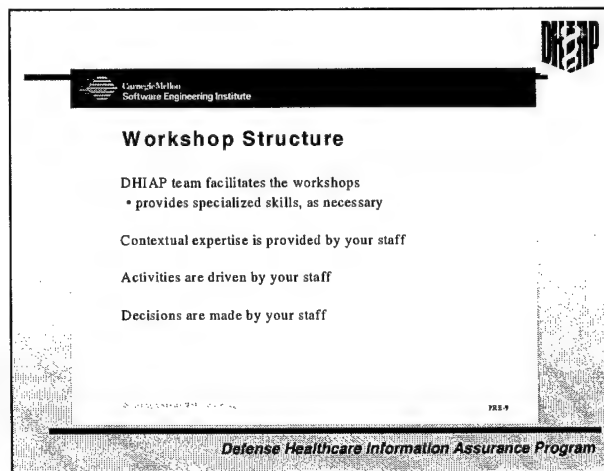
Proactive

Defense Healthcare Information Assurance Program

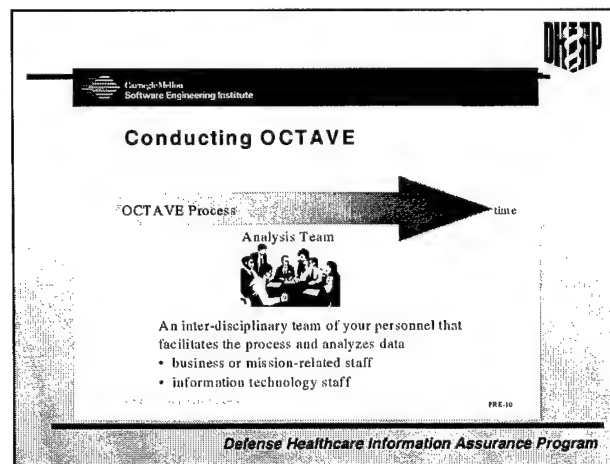
Slide 12



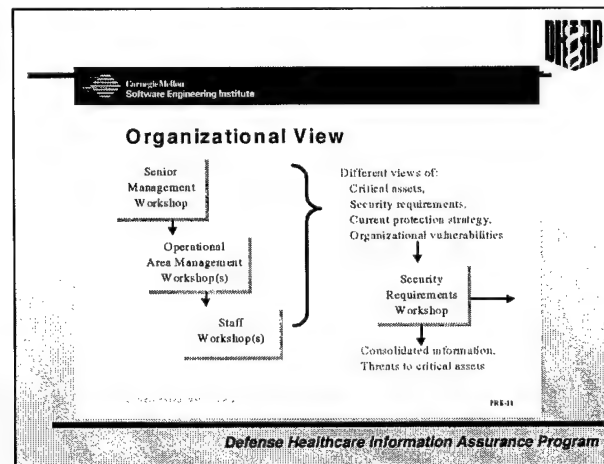
Slide 13



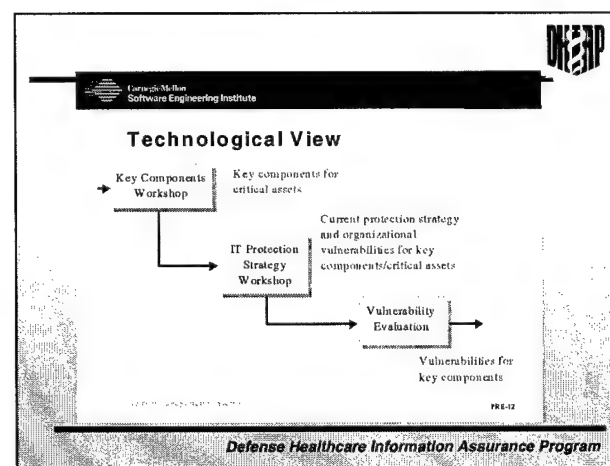
Slide 14



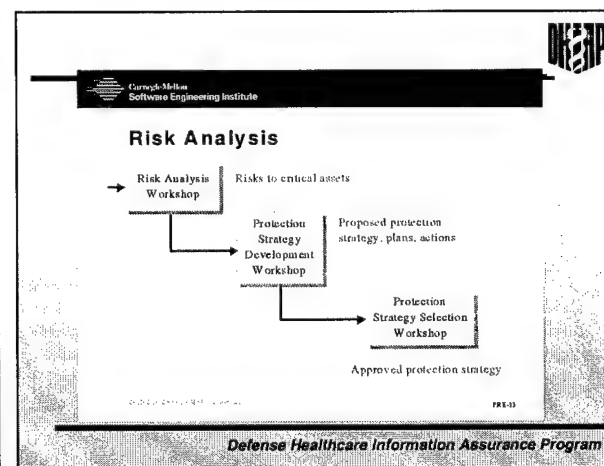
Slide 15



Slide 16

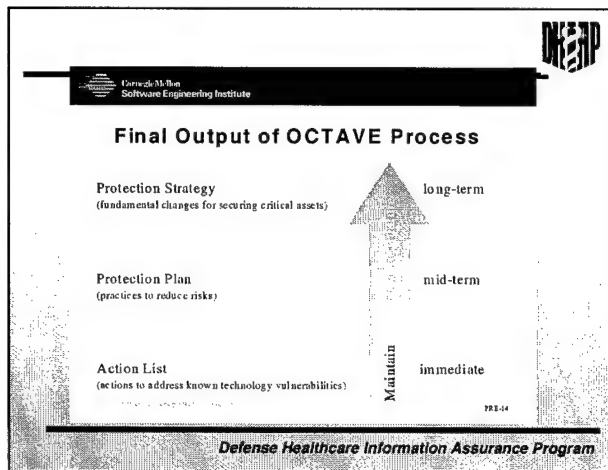


Slide 17

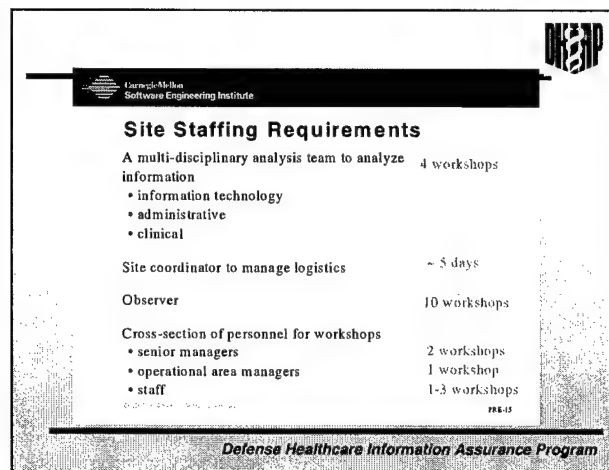


Slide 18





Slide 19



Slide 20

**Site Staffing Requirements -2**

Senior Management Workshop	Senior Managers
Participants Briefing	All participants
Operational Area Management Workshop(s)	Operational Area Managers
Staff Workshop(s)	Staff
Security Requirements Workshop	Analysis Team
Key Components Workshop	IT & Business Staff

PRE-16

Defense Healthcare Information Assurance Program

Slide 21

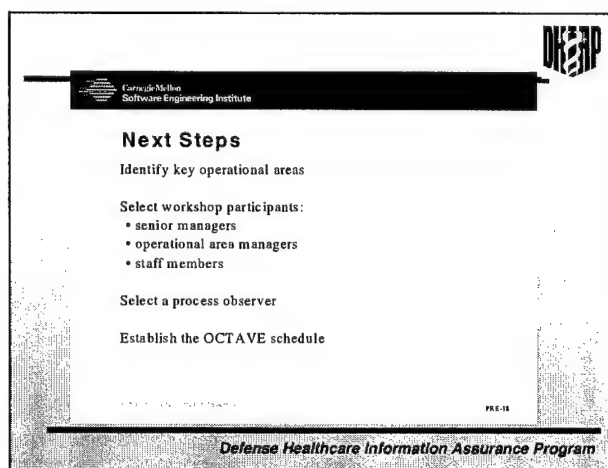
**Site Staffing Requirements -3**

IT Protection Strategy Workshop	IT Staff
Vulnerability Evaluation	IT Staff
Risk Analysis Workshop	Analysis Team & selected staff
Protection Strategy Development Workshop	Analysis Team & selected staff
Protection Strategy Selection	Senior Managers, & Analysis Team
Closing Briefing	All Participants

PRE-17

Defense Healthcare Information Assurance Program

Slide 22



Slide 23

**BCA #2 Scope**

1. Title: Effective Authentication in a Medical Environment
2. Purpose: In the context of MTFs, analyze capabilities and limitations of various authentication technologies, in light of MTF situational dependencies, and present results and indicative costs for acquisition decision-makers.
3. Why:
  - Current authentication techniques are:
    - Inadequate
    - Prone to abuse
    - Subject to penetration
    - Difficult to manage
  - Emerging technology can provide improved solutions for authentication and reduced vulnerabilities
4. Audiences:
  - Acquisition staff for medical community
  - Sponsor
5. Timeframe: 60 days (see Draft schedule)
6. Cost: TBD
7. Type of Analysis: "Scenario-dependent technology applicability study"
8. Functional (User) Target Environment
  - MTF users (local and remote; i.e., MTF work sites such as nurses station, pharmacy, radiology, OR, ER, etc.)
9. Technical Target Environment / Current Plan
  - 1<sup>st</sup> Tier: Current Environment – NT, CHCS, network
  - 2<sup>nd</sup> Tier: CHCS II, Win 2000
  - 3-year horizon
  - Validation of User ID – yes
  - Authentication of source information – no
10. General Approach (How, Intended Methodology)
  - Build scenarios
  - Evaluate technology
  - Map the two together
  - Develop conclusions and recommendations
  - Write report

11. Potential participants

- Vendors
- Army Biometric Center
- CHCS II Development Team
- LAFB / WACH / DDEAMC

**BCA #3 Scope**

1. Title: Role-Based Access Control (RBAC)
2. Purpose (Goal) - In the context of the MTFs:
  - Analyze the requirements for role-based access to patient-identifiable medical information and medical/healthcare information systems relative to
    - HIPAA
    - Other regulatory guidance
  - Identify existing capabilities and planned approaches to satisfying those requirements
    - CHCS
    - CHCS II
    - Other MTF systems containing patient information not included in CHCS/CHCS II
    - Network access controls
  - Technical and operational impacts of implementing RBAC
3. Potential Barrier
  - Need access to CHCS and CHCS II sites and staff
  - Need access to CHCS II Design Team
4. Why (Threat, Risk, etc.)
  - Current information systems are inadequate due to
    - Inadequate user authentication
    - Inadequate application of user privilege policy
    - Inability of existing information systems to control access
  - Requirements for role-based access may change with implementation of HIPAA and/or other regulatory guidance
5. Audience(s)
  - MTF
    - Commander
    - IMD/IMO
    - ISSO
    - System Administrators of affected systems
  - Acquisition staff for medical community
    - Medical Material Acquisition Officer
    - At TATRC, at OSD/HA
  - Sponsor
6. Timeframe: 90 days
  - Background information due in four weeks
  - Conference (call) two days later:
7. Type of Analysis:
  - RBAC Gap analysis: Requirements vs. technical environment (current & planned)
  - Impact analysis of RBAC technology at an MTF/regional facility

8. User Functional Target Environment
  - MEDCOM ISSO
  - MTF ISSOs
9. Technology Target Environment / Current Plan
  - MTF information environment
  - CHCS
  - CHCS II
10. How / Intended Methodology / Approach
  - Understand HIPAA role-based requirements
  - Understand other policy requirements, i.e., JCAHO
  - Understand CHCS and CHCS II role-based access
  - Understand other patient information repositories
  - Survey/interview system administrators
  - Map all of the above
  - Identify shortfalls
  - Develop conclusions and recommendations
  - Write report
11. Participants
  - CHCS Program Office
  - CHCS II Program Office
  - HIPAA implementation authority
  - MTF and MEDCOM system administrators
  - DHIAP BCA Team
12. Major Activities

**BCA #4 Scope:**

1. Title: Network Audibility (NA)
  - Purpose (Goal) - In the context of the MTFs:
  - Analyze the requirements for auditing access and movement of patient-identifiable information relative to
    - HIPAA
    - Other regulatory guidance
  - Identify existing capabilities and planned approaches to satisfying those requirements
    - CHCS
    - CHCS II
    - Other MTF systems containing patient information not included in CHCS/CHCS II
    - Network access controls
  - Technical and operational impacts of implementing AA
2. Potential Barrier
  - Need access to CHCS and CHCS II sites and staff
  - Need access to CHCS II Design Team
3. Why (Threat, Risk, etc.)
  - Requirements for access audibility may change with implementation of HIPAA and/or other regulatory guidance
  - Current information systems are inadequate due to
    - Inadequate access audibility
    - Inadequate application of access audibility policy
    - Inability of existing information systems to compile and provide aggregate logs
4. Audience(s)
  - MTF
    - Commander
    - IMD/IMO
    - ISSO
    - System Administrators of affected systems
  - Acquisition staff for medical community
    - Medical Material Acquisition Officer
    - At TATRC, at OSD/HA
  - Sponsor
5. Timeframe: 90 days
  - Background information due in four weeks
  - Conference (call) two days later:
6. Type of Analysis:
  - NA Gap analysis: Requirements vs. technical environment (current & planned)
  - Impact analysis of NA requirements at an MTF/regional facility

7. User Functional Target Environment
  - MEDCOM ISSO
  - MTF ISSOs
8. Technology Target Environment / Current Plan
  - MTF information environment
  - CHCS
  - CHCS II
9. How / Intended Methodology / Approach
  - Understand HIPAA NA requirements
  - Understand other policy requirements, i.e., JCAHO
  - Understand CHCS and CHCS II NA capabilities
  - Understand AA of other patient information repositories
  - Survey/interview system administrators
  - Map all of the above
  - Identify shortfalls
  - Develop conclusions and recommendations
  - Write report
10. Participants
  - CHCS Program Office
  - CHCS II Program Office
  - HIPAA implementation authority
  - MTF and MEDCOM system administrators
  - DHIAP BCA Team
11. Major Activities

**Survivability and Simulation**  
**David A. Fisher**  
**Software Engineering Institute**

**Abstract.** The past few years have shown a need for fundamentally new approaches to the security and survivability of large scale networked systems. Infrastructures and other modern systems pose problems that arise from many factors: partial information, complexity of combined interactions of very large numbers of human and automated participants, ubiquitous access, loss of centralized administrative control, the growing threat of automated attacks, and a shift in priorities for security from confidentiality of information to availability of services. Survivability research seeks new methods, including emergent algorithms, diversity, and dynamic trust validation, to ensure that systems satisfy their most critical requirements. Easel is an automated simulation tool for research in survivability, infrastructure assurance, and other applications that must contend with incomplete and imprecise information.

**Introduction.** Survivability is a relatively new research domain that seeks new techniques for mission fulfillment in unbounded systems such as the Internet, electric power, telecommunications, military command and control, and health care infrastructure. *The role of infrastructure assurance is to ensure the survivability and fulfillment of these missions. It is about managing business risks to satisfy critical mission requirements in the face of competition and adversity. It is about making trade-offs to ensure continuity of services in the absence of complete and precise information.*

**Survivability.** Survivability is defined as the ability of a system to fulfill its mission in the presence of attacks, failures, and accidents. Survivability and infrastructure assurance are *not* about adherence to industry standard security practices. They are about confidentiality, information integrity, correctness, and dependability only to the extent that these particular quality attributes are the most critical requirements of the mission. Successful infrastructure assurance depends on sound business decision making and cannot be achieved through technological solutions alone.

Security technologies are often effective when

- the primary objective is confidentiality or integrity of information
- the system has a clearly defined perimeter with centralized administrative control
- administrators have complete insight into all aspects of the system
- all components and participants within the system are known, authenticated and trustworthy

Perhaps most important is the assumption that applications can tolerate a binary fortress model of security in which either all attempted intrusions are



successfully repelled or the system as a whole is compromised. Security solutions are inadequate if even a few of these conditions are absent. Unfortunately, modern large scale networked systems, including critical national infrastructures, seldom satisfy any of these conditions.

**Unbounded Systems.** An unbounded system is any system in which the participants (human or computerized) have only incomplete or imprecise information about the system as a whole. They include human participants as well as automated components. Their boundaries are not precisely known. Interconnections among participants in unbounded systems change constantly. Furthermore, the trustworthiness and often the identity of participants is unknown. Centralized administrative control cannot be fully effective in such systems. These are the characteristics critical national infrastructures, of the Internet, and of electronic commerce. They characterize most social, economic, and biological systems, and most activities one participates in every day. Such systems contrast dramatically with the assumptions of closed, centrally-controlled computer systems and with the assumptions underlying many modern computer security technologies.

**Security Technology.** Technologies based on fortress models of security are successful in transparent, trusted, centrally-controlled systems with clearly defined perimeters because they exploit those constraints in their environment. Success in survivable systems and infrastructure assurance likewise depends on exploiting the properties of unbounded systems to full advantage. It is unlikely that any large networked system can be molded to satisfy the constraints of fortress based security. In contrast with traditional security methods that attempt to build impenetrable walls around trusted insiders, a fundamental assumption of survivability is that no participant or component of a system is immune to all errors, failures, and compromises.

**Survivability Technology.** Survivability research seeks solutions that exploit the inherent characteristics of unbounded systems to ensure survivable missions in the presence of compromised components. Unbounded systems offer opportunities for cooperation without the cost and vulnerabilities of coordination. They enable distributed specification and local optimization with their attendant resistance to standardized attacks. They facilitate robust and resilient solutions in which no component is essential and compromises of individual components will not cascade.

A survivability architecture is an abstraction that specifies the critical characteristics required for mission survivability in an unbounded system. A survivability architecture consists of the local actions of each node and the protocols of interactions among neighbor nodes. Any algorithm that fulfills mission critical requirements by exploiting the characteristics of an unbounded system is an emergent algorithm. Emergent algorithms instantiate survivability architectures for specific mission requirements.

**Emergent Algorithms.** Emergent algorithms differ from conventional hierarchical and distributed algorithms; they operate in the absence of complete and precise information, do not have central control, hierarchical structure, or other single point vulnerabilities, and they achieve cooperation without coordination. Mission requirements are satisfied in the form of global system properties that emerge from the combined actions and interactions of all system components. For reasons of mission survivability, our research considers only emergent algorithms that do not have single (or any fixed number of) points of failure.

For reasons of practicality and affordability, we consider only those emergent algorithms in which the cost of each node (whether measured in dollars, CPU cycles, storage requirements, or communication's bandwidth), is less than proportional to the number of nodes in the system. The effectiveness of this approach can be further enhanced by dynamic trust validation among the participants.

**Need for Simulation.** Although the benefits of ad hoc development of emergent algorithms have been demonstrated, a rigorous process for deriving emergent algorithms from mission requirements is prerequisite to their widespread use in automated systems. Our intuitions about the global effects of the local actions and interactions among large numbers of nodes are seldom correct. The problem of designing emergent algorithms is especially difficult, because it begins with the desired global properties and attempts to determine what simple combinations of local actions and interactions would produce those effects over time in a large scale network. An effective design methodology will depend on greater understanding of the influences of local action and interaction on emergent global properties and on the sensitivities of emergent properties to local variations. The obvious (and probably only) means to answer these questions is by simulation of emergent algorithms and of the unbounded systems in which they operate. This recognition has opened a new area of research for simulation of unbounded systems.

**Limitations on Accuracy.** Current simulation systems do not produce accurate predictions of the behavior of unbounded systems. By definition, unbounded systems are incompletely and imprecisely defined. Thus, a simulation of an unbounded system must be able to produce accurate results based only on incomplete information. Current models, however, require complete information and thus are always built with assumptions or inaccurate information. The ability to operate on abstract specifications and simulate at various levels of abstraction is a long-standing need of many applications, but is not provided as a feature of existing simulation systems.

Equally important, all object-based models (both physical and computerized) are inherently inaccurate because they are based on complete representations

as objects. This might be acceptable when dealing with small numbers of nodes or when great care is taken to differentiate between which modeling results are likely to be valid. Such remedies seldom, if ever, succeed in differentiating inaccurate results when modeling complex or large scale systems. Furthermore, as the number of subsystems in a model increases, the inaccuracies of each subsystem pervades the whole after a few iterations and guarantee that all simulation results will be inaccurate. This may account for the pervasive failure of large scale simulations to produce accurate results. These problems are aggravated in unbounded systems where the numbers of components are very large and a primary purpose of simulation is to accurately predict the global effects of local activities. Because accuracy and completeness are not simultaneously achievable when describing the physical world, accurate simulation is feasible only if the simulator can guarantee accurate results from accurate but incomplete specifications.

Other difficulties in simulating unbounded systems include

- the need for thousands to tens of thousands of nodes per simulation
- the lack of linguistic mechanisms in programming languages for making incomplete and imprecise specification
- the inability of object oriented computations to describe and reason about the real world
- the need to combine information about a system from multiple knowledge domains
- management of multiple simultaneous beliefs of the various stakeholders in an infrastructure
- integration among separately developed simulations
- exponential increases in computational cost that accompany linear increments in the granularity or number of nodes in a simulation.

**Easel Simulation System.** These considerations have led to a new approach to simulation called Easel, an emergent algorithm simulation language and environment. Easel employs a paradigm of property-based types (i.e. describing abstract classes of examples by their shared properties) to simultaneously address all of the above simulation problems. Because Easel is property-based it can be used to give accurate, but incomplete, descriptions of anything. In combination with an appropriate automated logic system, it can be used to produce accurate conclusions about examples from the physical world. This contrasts with physical models and automated simulations that depend on representations of objects, where descriptions must be complete (and thus inaccurate), and in which conclusions are accurate only for the model but never for their extensional interpretation in the real world. While traditional modeling and simulation systems answer all questions without a mechanism for user to determine which answers are accurate, Easel reports what additional information is needed to continue

toward an accurate result. Easel also supports multiple levels of abstraction, multiple simultaneous belief systems, distributed specification, and dynamic graphic depictions.

Easel is a discrete event simulation language but adds limited support for a continuous variables. The linguistic limitations of traditional programming systems for incomplete and imprecise description are overcome by the use of quantifiers, adjectives, improper nouns, pronouns and other forms of anonymous reference in the language. In combination with property-based types, these mechanisms provide a semantic framework for referring examples of any type, whether real or imagined, and whether from the computational, mathematical, or physical worlds.

**Development Status.** Easel is currently under development. Preliminary versions of the Easel Language Reference Manual (ERLM) and the Easel Author Guide (EAG) are scheduled for release with the alpha version of the system at the end of February 2001. A working version that includes the translator, interpreter, graphic depiction system, and run-time support (but with an incomplete set of operations) is now being tested. The beta release is scheduled for September 2001. Until then, much work is required to validate the language design for a few of the most important applications, to complete the system design and implementation, and to measure and tune system performance. We anticipate that additional requirements will be discovered during this process.

**Research and Development Partners.** External partners and sponsors are needed for continuing Easel development and applications. Resident affiliates are sought for research in survivability, simulation, and emergent algorithms. Partners are also sought for experimental development of applications in infrastructure assurance and other unbounded domains. Except for research and development partners as described above, we do not anticipate beta sites for Easel outside Carnegie Mellon University.

**Summary.** Survivability requires new techniques for mission assurance in the absence of complete and precise information. Promising approaches include using emergent algorithms for resilient and robust systems in the presence of multiple component compromises, dynamic trust validation for marginalizing compromised components and participants, and cooperation without coordination for avoiding single points of vulnerability. The Easel simulation system holds promise as an effective research tool for fulfillment of critical mission requirements in infrastructures and other applications that must operate with incomplete information. Easel also overcomes several long-standing barriers to accuracy, scaleable, and abstraction in large-scale simulations.

Copyright © Carnegie Mellon University 2000



# Easel Language Reference Manual

Version 1.0.3.3 October 10, 2000

## DRAFT REVIEW COPY

This version of the ELRM is a draft. It is believed to be adequately complete and correct for the alpha and beta implementations of the Easel simulation system. Comments are requested. Although substantive changes to the language design and ELRM content are not anticipated prior to the alpha release, continuing revisions will be made for issues that arise during the ERLM review process and the implementation effort. Work will also continue on sections currently marked as incomplete as well as corrections in spelling, grammar and presentational style. Note: markings showing implementation plan, implementation status and ELRM status have not been removed from this version.

NOTE: ELRM subsections that are only under consideration for futuer versions and will not be available in the beta version have their section numbers enclosed in [ ].

Software Engineering Institute  
4500 Fifth Avenue  
Pittsburgh, PA 15213

# Language Reference Manual

## Content:

- 0. Introduction
- 1. Types
- 2. Examples
- 3. Ordered and Scalar Types
- 4. Numbers
- 5. Attributes
- 6. Operations and Statements
- 7. Actors, Neighbors and Simulations
- 8. Dynamic Graphic Depictions
- 9. Persistent Data
- 10. System Level Features
- 11. Example Programs
- A. Syntax and Grammar
- B. Messages and Reports
- C. Running and Controlling Simulations
- D. Debugger
- E. Graphic Editor
- F. CGI Interface

- 0. Introduction
- 1. Types
  - 1.1. Property Based Types
  - 1.2. Lambda Types
  - 1.3. Operations on Types
  - 1.4. The Universal Type
  - 1.5. Booleans
  - 1.6. Dimensional Analysis
  - 1.7. Units of Measure
  - 1.8. Representation Types
- 2. Examples
  - 2.1. Defining and Referencing Examples
  - 2.2. Pronouns
  - 2.3. Quantifiers
  - 2.4. Lambda Application
  - 2.5. Adjectives
- 3. Ordered and Scalar Types
  - 3.1. Ordered Types
  - 3.2. Scalar Types
  - 3.3. Enumeration Types
  - 3.4. Machine Integers
  - 3.5. Power Sets
- 4. Numbers
  - 4.1. Floating Point Numbers
  - 4.2. Basic Numeric Operations
  - 4.3. Integer Conversion Operations
  - 4.4. Trigonometric and Hyperbolic Functions
  - 4.5. Statistical Functions
- 5. Attributes
  - 5.1. Attributes Definitions
  - 5.2. Operations on Attributes
  - 5.3. Composite Types
  - 5.4. Mutable Types
  - 5.5. Sequences and Arrays
  - 5.6. Lists
  - 5.7. External Attributes
- 6. Operations and Statements
  - 6.1. Defining Operations
  - 6.2. Sequential and Conditional Control
  - 6.3. Iterative Control
  - 6.4. Interpretation Semantics
  - 6.5. Mobile Code and Expressions as Data
- 7. Actors, Neighbors and Simulations
  - 7.1. Actors
  - 7.2. Neighbor Relationships
  - 7.3. Simulations
  - 7.4. Belief Systems
  - 7.5. Simulated Time
  - 7.6. Continuous Simulations
  - 7.7. Observers and Facilitators



- 7.8. Queuing.
- 8. Dynamic Graphic Depictions
  - 8.1. Views and Windows
  - 8.2. Drawings
  - 8.3. Groups
  - 8.4. Regions
  - 8.5. Operations on Regions
  - 8.6. Patterns
  - 8.7. Text
  - 8.8. User Input
- 9. Persistent Data
  - 9.1. Whole File Operations
  - 9.2. Sequential File Operations
  - 9.3. Formatting and Printing
  - 9.4. Folders
  - 9.5. Persistent Storage
  - 9.6. Version, Configuration and Release Control
  - 9.7. Network Communications
- 10. System Level Features
  - 10.1. System Attributes
  - 10.2. System Root
  - 10.3. Translator
  - 10.4. Interpreter
  - 10.5. Memory Management
  - 10.6. Windows
  - 10.7. Event Handling
  - 10.8. Foreign Code Interface
- 11. Example Programs
  - A. Syntax and Grammar
    - A.1. Lexical Grammar
    - A.2. Semantic Grammar
    - A.3. Parse Grammar
    - A.4. Meta Grammar
  - B. Messages and Reports
    - B.1. Error Messages
    - B.2. Assumptions
  - C. Running and Controlling Simulations
  - D. Debugger
  - E. Graphic Editor
    - E.1. Mouse Operations
    - E.2. Keyboard Commands
    - E.3. Draw Menu
    - E.4. Shape Menu
    - E.5. Paint Menu
    - E.6. Text Menu
  - F. CGI Interface

## 0. Introduction

Easel is a programming language for simulating systems for which only incomplete or imprecise information is available. It allows abstract descriptions of anything. The things described may be real or imaginary. They may be physical, abstract, or computational. Easel is intended as a research tool for describing and simulation unbounded systems, emergent algorithms and survivability architectures.

Easel is a property-based language. It computes on (accurate but incomplete) abstractions in the form of property based types.

## 1. Types

### 1.1. Property Based Types

Property Based Types (PBT) is a system for describing categories of things by their properties. This contrasts with object oriented systems which describes classes by their examples (or more precisely, by representations of their examples). In a PBT system, a type is a set of properties. Each property is a predicate that is true for every example of the type. Anything that satisfies the properties of a type is an example of that type. A type may describe examples that are physical, such as people, for or the earth; that are abstract such as integers, greed or beliefs; or that are computational such as machine integers, data structures or the content of a disk. Types are the primary objects of computation in Easel.

More formally, a property based type is the transitive closure of a set of properties under implication in an intensional logic. Types provide a formal mechanism for accurately representing and reasoning about abstractions, informal ideas and incomplete and imprecise specifications.

Types are the primary objects of computation in Easel. Types are neither sets of objects nor data representations. Each type is incompletely described and thus may have unmentioned properties. The names of type are used like improper nouns in natural languages. Because any set of properties defines a type, there are an infinitude of types. Most types do not have names and instead are described by their differences (e.g., adding adjectives) from named types using a variety of operations including abstraction and specialization.

If *s* is a *subtype* of type *t*, then every property of *t* is also a property of *s* and every example of *s* is also an example of *t*. That is, a subtype has a subset of the examples and a superset of the properties of the any type of which it is a subtype.

#### 1.1.0. Syntax

`body ::= "{" [ { prop ";" } ] "}"`

# body of type definition

`prop ::= "property" exp`

# predicate or supertype property

#### 1.1.1. type: immutable type;

#. type type

"Type" is the type of all types including itself. Every type is an example of type type. Every type is also a subtype of type type. Each type is an immutable abstraction.

#### 1.1.2. "body"( property... ): type;

#. type body

The body operation encloses the set of properties defining a type in a type definition as described in section 1.1. Body may contain any number of properties. Each parameter to body must be a property. Attribute definitions (see section 5.1.1) and property assertions (see sections 1.1.3 and 1.1.4) must appear before the statements (see chapter 6). Attribute initialization and statements will be evaluated in the order of their

appearance. Property assertions are noted and taken as true throughout the most local enclosing simulation throughout the life of the simulation. Syntactically, a type body must be enclosed by "{" "}" or by line indenting (see section A.1.8).

1.1.3. `property( exp type): property;` #. supertype property

A supertype property asserts that all properties of the given type are also properties of the current type. That is, the given type is a supertype of the current type. Often the supertype will be an anonymous type specified as a named type specialized (i.e. the "&") by some predicate. The predicate may reference the pronoun "it" (see Section 2.2.1) meaning any arbitrary value of the supertype.

1.1.4. `property( exp boolean): property;` # predicate property

A predicate property asserts that the boolean expression is true throughout the current simulation or belief system. A predicate property, however, may reference the pronoun "this" (see Section 2.2.2) meaning any arbitrary example of the type currently being defined.

### 1.1.5. Other Property Specifications

There are several other forms of property specification including attributed properties of each example of a type (see Section 4), relational properties among types and their examples (see Section 6.1), operational properties that describe the activities of types with dynamic examples (see Section 6.XXX), asserted properties that specify state relationships that prevail during these activities ( see Section 6.XXX), autonomous properties of actors ( see Section 7), neighbor relationships and interactions among actors (see Section 7.3), explicitly inherited properties from other scopes (section XXX) and libraries (section 1.1.6 and 1.1.7).

1.1.6. `include( string | file): any type;` #. include property

The include operation reads the content of a file or record into the program at compile time. The file may be specified either by the string name of the file in the underlying operating system or as a Easel file or record specification that is compile time evaluable. The designate file or record must be a string that is a syntactically well formed structure of an Easel program. Include would typically be used to include definitions or drawings.

## 1.2. Lambda Types

Any named type may be parameterized. Several parameterized types may have the same name provided they have different numbers of parameters, have different formal parameter types, or produce the same results when applied to the same actual parameters. In some contexts they can also be distinguished by their return types.

### 1.2.0. Syntax

`fpl ::= "(" [ { fpl "," } ] fpl [ "..."] ")"` # formal parameter list  
`fp ::= [ id ":" ] type` # formal parameter

1.2.1. `lambda( list type, rt: type): type;` #. lambda type

Any definition may be parameterized. Lambda is the type of all parameterized types and operations. The first parameter to lambda is the formal parameter list of the type or operation. The second parameter is the return type. The specification "`f( x: t; y: s): u;`" is a short hand for "`f: lambda( [ x:t; y: s], u): type;`". The type of a

parameterless operation is `lambda( nil, ?)`. Syntactically, the formal parameters of a lambda definition are surrounded by parentheses and placed between the name and semi colon of the define. The type specified after the colon in the type definition becomes the rt of the lambda and the lambda type is the second parameter to the define.

1.2.2. `"fp_spec"( identifier, type): type; #. formal parameter specification`

A formal parameter is a special form of attribute. It is an attribute of both the lambda subtype and of all examples of that subtype. Formal parameter are unassignable unless explicitly specified as variable (see Section 5.1.7). A formal parameter specification gives the domain of actual parameter values for each formal parameter of a lambda type. Formal parameters may have identifier names. If a formal parameter is named, its value may be referenced by that identifier from within the formal parameter list and within the body of the lambda definition. The actual parameter value may also be referenced by dot qualification on the lambda type or any example of the lambda type. By convention, if a lambda definition gives special treatment to a subtype of the type of a formal parameter, the formal parameter type is specified as a union of the subtype and the type.

1.2.3. `"..."( type): type; #. variable number of parameters`

The variable number of parameters suffix is applied to the type of the last formal parameter. Any number (i.e. zero or more) of actual parameters may correspond to a variable number of parameters formal parameter.

1.2.4. Referencing Variable Numbers of Parameters  $\beta$

The actual parameters corresponding to a `"..."` formal parameter (see section 1.2.3) are collected into an array whose components may be referenced by indexing the formal parameter name (see Sections 5.3.5 and 5.4.4).

1.2.5. `"apply"( lambda, any type... ): lambda type; #. partial application  $\beta$`

Partial application is syntactically similar to application, but has one or more unspecified actual parameters (see section 1.4.1). Instead of returning the specified result or action, a partial application returns the anonymous lambda type created by substituting the specified actual parameters for their corresponding formal parameter names in the body of the lambda definition.

1.2.6. `lambda( fpl: list type, rt: type, body: exp):  $\gamma$  #. lambda expression`  
`lambda( fpl, rt) type; #.`

An anonymous lambda type may be defined at the point of its application. This form of lambda has three parameters: the formal parameter list, the return type, and the body. No define is required. This form corresponds closely to the anonymous lambda of recursive function theory.

### 1.3. Operations on Types

Most types do not have names. One way to create anonymous types is to compute them using type valued operations.

#### 1.3.0. Syntax

`op ::= "<<" ">>" "&" "|" "!" "isa" "~"` # type operators

1.3.1. `"&"( type, type): type; #. specialization`

The “&” operation, read as “and”, specializes a type to the properties of another type. This is a commutative and associative operation. The resulting type is the union of their properties and the intersection of their examples. The result is the least restrictive common subtype of the arguments. Any operation that applies to examples either of the argument types also applies to examples of the result type. If a boolean expression is used where a type is required, it is interpreted as a type consisting only of that property. The latter form is commonly used as the second argument to “&”.

1.3.2. “~”( type, type): type; #. abstraction

The abstraction operation returns the intersection of the properties of its arguments. Abstraction is commutative and associative. The abstraction of two types is a supertype of their union type. Any operation that applies to examples of the result type also applies to examples of both argument types.

1.3.3. “|”( type, type): type; #. type union

The type union operation, read as “or”, returns the type which is the union of the examples of its arguments. This is a commutative and associative operation. The result is a supertype of the arguments, but its examples do not necessarily have any shared properties. Any operation that applies to examples of the result type also applies to examples of both argument types.

1.3.4. “~”( type): type; #. unary abstraction

The unary abstraction operation returns the intersection of the properties of all of the examples of its argument. Thus, for any two types  $t1$  and  $t2$ ,  $(t1 \sim t2) = \sim(t1 | t2)$ . Unary abstraction is also useful for obtaining the type of an example.

1.3.5. “|”( type): type; #. type complement

The type complement operation, read as “not”, returns the type of all examples which are not examples of its argument. The specialization of the argument with the result is the inconsistent type. The abstraction of the argument with the result is the universal type. The union of the argument with the result is the universal type.

1.3.6. diff( a: type, b: type): type; #. type difference

The difference operation returns all properties of the first argument that are not properties of the second parameter. This operation is particularly useful for determining the similarities among examples of a mutable type.

1.3.7. “<<”( type, type): boolean; #. subtype predicate

The subtype predicate returns true iff the first argument is a subtype of the second. If neither is a subtype of the other it returns false.

1.3.8. “>>”( type, type): boolean; #. supertype predicate

The supertype predicate operation returns true iff the first argument is a supertype of the second. If neither is a supertype of the other it returns false.

1.3.9. “isa”( all, type): boolean; #. is an example of

The isa operation returns true iff the first argument is an example of the second.

#### 1.4. The Universal Type

The universal type is the type of no properties and all examples. It is a supertype of

all types and a subtype of only itself. Formally, it has no properties that are not tautologies. The quantifier “all” may be used to reference the universal type. The quantifier “any” may be used to reference any arbitrary example of the universal type. The universal type may also be referenced, and usually is throughout this manual, as “any type”.

#### 1.4.0. Syntax

<code>exp ::= "?"</code>	# unspecified value
<code>  "(" exp ")"</code>	# parenthesized expression
<code>op ::= "="   "!="</code>	# equality operators

#### 1.4.1. “?”: any type;

#. unspecified

The pseudo value, “?”, indicates that a value of an expression or action is unspecified at the point of the “?”. The unspecified value is a mechanism for incomplete and imprecise specification, for avoiding redundant specifications, and for delaying specifications. The unspecified value may be used anywhere an expression or statement is allowed. The effect of “?” depends on the context of its use. In its most general use, it is a form of incomplete or imprecise specification and indicates that the information is unknown, unavailable or unneeded. The interpretation of unspecified depends on the context of its use. As the name in a definition (see section 2.1.1) or attribute definition (see section 5.1.1), unspecified indicates that the definition or attribute is anonymous. As the body of a define (see section 2.1.1 and 6.1.1), it indicates that the definition is incomplete. As the initial value of an attribute (see section 5.1.3), it indicates that the initial value is not specified at that point. As a statement (see section 6.1.6), it indicates that there may be intervening unspecified statements. As the value of an actual parameter to an operation or parameterized type (see section 1.2.5), it returns a subtype of the type with the specified parameters instantiated.

#### 1.4.2. nil: any type;

#. nil value

Nil is a pseudo value of any type. It indicates an error, inconsistency, or absence of a value. It may be used anywhere that an expression or statement is required. Nil may be assigned to and referenced from any attribute, passed as a parameter, or returned as the value of an operation. If an error or inconsistency occurs during evaluation of an operation with a nil parameter, the error is not reported and instead nil is returned as the value of the operation. This allows programs to continue execution without cascading error reports. If a routine accepts nil as a valid actual parameter value, nil must be included in the formal parameter type.

#### 1.4.3. “()”( any type): the type;

#. identity

Expressions must sometimes be enclosed in parentheses to override the operator bindings built into the language, to enclose aggregates as arguments to single argument lambdas, or for clarity. Parentheses are interpreted as an identity function on examples of any type.

#### 1.4.4. “=”( any type, any type): boolean;

#. equality

The equality operation returns true iff its arguments are references to the same example. Equality may be applied to (references to) any two examples regardless of their types.

#### 1.4.5. “!="( any type, any type): boolean is !x = y;

#. inequality

The inequality operation returns true iff its arguments are not references to the same example. Inequality may be applied to any two examples regardless of their types.

## 1.5. Booleans

### 1.5.0. Syntax

`exp ::= "for" [ id ":" ] exp "do" exp` # quantified condition

#### 1.5.1. boolean: type is type;

#. boolean type

Boolean is a type with two truth values: true and false. Truth is the property of being in accord with some belief system. Any type can be viewed as a belief system in which the properties are the beliefs. Thus, to ask whether a boolean expression is true at some point in a program, is to ask whether it expresses a property that is consistent with the current type or belief system at that point.

#### 1.5.2. true: boolean;

#. true

True is a truth value that is consistent with the current belief system. That is, it is the type of all consistent types.

#### 1.5.3. false: boolean;

#. false

False is a truth value that is inconsistent with the current belief system. That, it is the type of all inconsistent types.

### 1.5.4. Operations on Booleans

`!"( type): boolean;`

#. boolean not operation

Because true and false are types they inherit all of the operations defined on types (see Sections 1.3 and 1.8). When the binary type operations `&` and `|` are applied to types the produce results with the intended boolean interpretation. Unary `|` however yields the correct boolean interpretation however only when applied to true or false. Thus, the boolean not operation `!` is provided. When applied to any consistent type including true, false is returned. When applied to an inconsistent type, not returns true.

#### 1.5.5. "for"( identifier, $\beta$

# quantified condition

`exp, exp boolean): boolean;`

#

A quantified condition is analogous to quantification in mathematics. The first argument is an identifier which is a free variable of the third argument. The second argument may be a type quantified by "any" or "some" corresponding to universal and existential quantification respectively. Alternatively the second parameter may be a list, enumeration, actor or prop quantified by "each" or "every" quantifier as used in iterative statements (see Section 6.3). In the case of quantified types, the examples of the type need not be enumerable. The value of a quantified condition in the case of "any", "each" and "every" is the "and" of the values of the third parameter when applied to every example. In some case, it is the value of the "or" of the values of the third parameter when applied to every example. Quantified condition may be used anywhere boolean expressions are allowed including as the condition of a property declaration (see Section 1.1.4).

## 1.6. Dimensional Analysis $\beta$

A dimension is any fundamental property such as mass, length or time on which measurement is possible. Dimensional analysis is a processing for verifying the



consistency of the dimensions used in numeric computations, such that for example only distance squared and not distance may be used as a measure of area. The author must specify the dimensionality of each formal parameter and of the return type of each operation. The translator will verify that each actual parameter in an application has the required dimensionality and that the operation produces a result of the specified dimensionality. Note that many mathematical functions are dimensionless.

1.6.1. dimension: type;

#. dimensions

Dimension is the type of all dimensions. Distance, area and direction are defined in the example below. Time is defined in Section 7.5. The body of a dimension may be used to specify the relationships among dimensions.

Examples:

1.6.2. distance: dimension;

#. distance dimension

1.6.3. area: dimension is distance \*distance;

#. area dimension

1.6.4. direction: dimension;

#. direction dimension

## 1.7. Units of Measure

A unit of measure is a standard for measuring within a dimension. Each dimension must have at least one unit of measure and may have several for variations in scale or preferences for particular measuring systems. Measuring systems may be systematic (e.g. the metric system) or ad hoc (e.g., the English system).

1.7.1. unit( dimension): type;

#. units of measure

Each unit of measure is an example of the subtype of unit for its dimension. Unless otherwise specified (see Section 1.7.2), it is assumed that all measures are scalable from zero. That is, two meters are twice the distance as one meter. The system automatically assigns a constant to each unit of measure. The exact value of the constant is unimportant as long as the zero value corresponds to the origin of the dimension and that the relative values of different units are maintained. In practice, the values are normally assigned to match the first unit defined for a dimension.

Examples:

meters: unit distance;

#. meters

kilometers: unit distance is 1000\* meters;

#. kilometers

centimeters: unit distance is meters/100;

#. centimeters

feet: unit distance is 0.3048 meters;

#. feet

inches: unit distance is feet/12;

#. inches

1.7.2. unit( dimension, offset: number): type;

#. offset unit of measure

Some units of measure, such as those for temperature, normally are not scaled from zero. This form of unit specification accommodates units scaled from other origins. The offset amount is the number of units in the measure from absolute zero to the zero of the unit.

Example:

temperature: dimension;

Celsius: unit( temperature, 273.15);

Kelvin: unit( temperature) is Celsius -273.15;

Rankine: unit( temperature) is 1.8 kelvin;



Fahrenheit: unit( temperature, 523.67) is rankine +491.67;

## 1.8. Representation Types

All representations of programs and data are currently determined by the translator and interpreter and cannot be specified by authors. The system is however capable of supporting multiple representations for a given type. In fact, examples of type int are sometimes represented as 16 bit integers and sometimes as floating point with implicit conversions between them as needed. Subtypes with the same attributes but different allocations of attributes to formal parameters have multiple representations with implicit conversions among them.

### [1.8.1.] Representation Specifications

Each different representation for examples of a type is a representational subtype of that type. The choice multiple equivalent representations does will not alter the outcome of any correct computation, but may affect the performance. No mechanisms are currently provided for explicit specification or author control of representations.

### 1.8.2. implementation( exp action): action; #. implementation effect only

As a general rule the logic of the body of an actor is assumed to describe not only the effect of some real world entity but the means by which that real world entity achieves that effect. Any procedural description may be enclosed as the parameter to the implementation operation to indicate that only its effects and not its method describes the real world. This distinction can affect both human understanding of the program and the conclusions of any proof system.

## 2. Examples

An example of a type is anything that satisfies or conforms to all properties of that type. Thus, examples are not restricted to things that can exist or be accurately represented in a computer. Instead, examples are restricted to those things whose types can be accurately described or represented in a computer. Easel makes no attempt to represent examples, but instead provides extensive facilities for describing types and uses quantification to reference examples of any named or described type. Each reference refers to, means, or denotes a example, but does not encode the example.

An anaphoric reference is any grammatical form used to reference a type or example without use of its proper name. Such mechanisms include examples referenced by pronouns (Section 2.2) and quantifiers (Section 2.3), and types referenced using adjectives (Section 2.5), type valued computations (Section 1.3) and other forms of anonymous types.

### 2.1. Defining and Referencing Examples

Proper nouns are the names of examples. They include language defined literals, author defined literals, the names of examples of enumeration types, and any defined identifier. The visibility rules for proper nouns are the same whether a literal, type, operation or other example is named. All definitions define an example of some type.

#### 2.1.0. Syntax

prop ::= id ":" type [ "is" body ]	# definition
exp ::= id   num_lit   str_lit	# atomic expressions

#### 2.1.1. "define" ( identifier, t: type, exp t): property; #. define

The define operation associates a name with an expression of any type. The first parameter is the name being defined and must be an identifier. The second parameter is any type of the value of the expression. The third parameter is an expression for the value. The name will be visible throughout the most local enclosing simulation scope of the definition, except where hidden by a more local attribute of the same name. The type must be some type of every value of the expression. The expression is evaluated each time the name is referenced and the resulting value returned as the value of the reference. If the first argument is unspecified, the definition is anonymous. If the second argument is unspecified, there are no restrictions on the type of the result and is similar to specifying the result type as all. If the third argument is unspecified, the definition is incomplete.

[2.1.2.] value( any type): any type; #. value specification

Value is a pseudo operation that causes its argument to be computed once at the time of scope entry and there after referenced as a constant of that value. It is most often used as the third argument to a non lambda define (see Section 2.1.1), but may be used anywhere. When the corresponding formal parameter to a value expression is marked as exp, the effect is to inhibit the effect of the exp; this is valid only if the actual parameter to the value operation is itself of type expression.

### 2.1.3. Visibility of References

Defined names are visible throughout their most local enclosing simulation. That is, they are defined in a scope that is global to the entire simulation, regardless of where their definition appears within the simulation, and thus are called global definitions. If several global definitions have the same name, all are visible, and no conflict occurs if they have different signatures (i.e. name, number and type of parameters, and return type). If two global defines have the same or overlapping signatures, they are legal iff they have the same value or affect in their overlap region. It is not possible to globally define attributes in Easel, so no conflicts can arise. An attribute definition however does hide all more global defines of the same name, but only for the immediate scope of the attribute definition and in particular not for subsopes of the attribute definition. If, however, a references is made to the name of a local attribute but the specified type of the attribute is illegal for the context of the reference, then global defines of the name are visible.

2.1.4. local( property): property; #. local scope specification

Any define may be specified as local, meaning that it is visible only within the local scope of its definition, and within more local embedded scopes, but not in more global scopes. Local definitions are illegal in the scope of an attribute definition of the same name. Local defines also hide any more global definition or attribute definition of the same name.

2.1.5. "ref"( token): any type; # reference definition

The define reference operation returns all definitions of the name that are visible at the point of reference. Multiple simultaneously visible definitions of a name is common. The body of each definition defines a type. The reference operation returns the union of those types. Each type in the union is called a candidate. Subsequent processing may reduce the number of candidates in the union type, either by formal parameter resolution (section XXX), context resolution (Section XXX), or optimization selection (Section XXX). If no definitions of the name are visible, an error is reported

and the token itself is returned. The visibility rules guarantee that attributes and defines of the same name are never simultaneously visible. Syntactically, the appearance of the identifier name alone constitutes a reference.

## 2.2. Pronouns

A pronoun is a language defined identifier that can substitute for a noun or noun phrase when the referent can be understood from the context of the reference.

Pronouns have global visibility but local meaning. Additional pronouns named self, sim, prog and lang are defined in sections 6.1.3, XXX, XXX and XXX respectively.

2.2.1. pronoun( property): property; #. pronoun declaration

Pronoun specifications apply only to definitions. They make the definition visible at the language level. Each pronoun typically has other built in significance.

2.2.2. pronoun it: all;  $\beta$  #. it pronoun

It refers to any example of a computed type where the reference to it is within the expression which computes the type.

2.2.3. pronoun this: all; #. this pronoun

This refers to the body scope of the most enclosing definition.

## 2.3. Quantifiers

A quantifier is a language defined identifier that names an operation that returns a reference to an example given a type. Because quantifiers are syntactically more binding than operators, they are convenient for marking the beginning of actual parameters without the use of parentheses, especially when the quantification is applied to an anonymous type composed from adjectives.

### 2.3.0. Syntax

exp ::= quantifier [{ exp }] type # quantification  
quantifier ::= "any" | "some" | "the" | "all" | "each" | "every" | num\_lit

2.3.1. "any"( t: type): t; #. universal quantification

The any quantifier returns a reference to an arbitrary example of its argument type. It makes a worst case selection, if possible referencing an example that will later lead to an inconsistency. An error will be reported later if any example leads to an inconsistency. The any quantifier should be used only in situations where any arbitrary example will suffice.

2.3.2. "some"( t: type): t; #. existential quantification

The some quantifier returns a reference to some example of its argument type. It makes a best case selection, if possible referencing an example that will not later lead to an inconsistency. An error will be reported if every example leads to an inconsistency. The some quantifier should be used only in situations where some especially chosen example will suffice.

2.3.3. "the"( t: type): t; #. definite quantification

The definite quantifier returns the only attribute in the current context of the given type. The attribute need not be visible nor named (including the argument to null, see Section 6.2.2). An error will be reported if the choice is not unique.

#### 2.3.4. "all" (t: type): type is t;

#. whole type quantification

The all quantifier returns a reference to the type of all of the examples of the argument type. It selects all of the examples at once and thus is an identity operation on types. Its primary use is to obtain the syntactic benefits of quantification for anonymous types.

#### 2.3.5. Numeric Literal Quantifiers

A numeric quantifier returns a reference to some examples of its argument type. It makes a best case selection of the number of examples specified, if possible referencing only examples that will not later lead to an inconsistency. That is, 8 dogs is equivalent to some 8 dogs. An error will be reported if more than the specified number of examples of the argument type lead to an inconsistency. Like the all quantifier, the result of numeric quantification is generally interpreted as a type. It is common to quantify the result of a numeric quantification.

#### 2.3.6. Stand Alone Quantifiers

Any quantifier may be used alone without an argument. In such cases, the result is the same as if the quantifier had been applied to the universal type. For example, "all" is all of the universal type, and "any" is any example of the universal type.

#### 2.3.7. Computed Quantifiers



Numeric quantifiers may be computed. The interpretation of computed quantifiers is the same as numeric literal quantifiers. The syntactic binding of computed quantifiers however is that of function names rather than quantifiers. Often it will be necessary syntactically, to enclose the computed quantifier together with its argument in parentheses.

#### 2.4. Lambda Application

A lambda application is an instantiation of lambda type or a operation with actual parameter values and evaluation of the resulting subtype or operation instance. Generally, it is necessary only to specify the name of the type or operation and actual parameter values of the correct number and type to uniquely distinguish which lambda is to be applied. The operation or lambda type however may be ambiguously referenced (Section 2.1.5) or may be computed. The actual parameter expression may also be ambiguous. The precise disambiguation of application is given in this section.

##### 2.4.0. Syntax

exp ::= exp "(" ")"	# lambda application - zero args
exp exp	# lambda application - one arg
exp "(" exp { "," exp } ")"	# lambda application - multiple args

##### 2.4.1. "apply" (t: lambda, any type...): t.r;

#. lambda application

Application of a parameterized type to the required number of actual parameter values of the required types, returns the value resulting from executing the body of the lambda definition. The lambda type to be applied is the union of all currently visible definitions of that name. The set of candidate definitions however is reduced by the following rules:

- eliminate all candidates that are not lambda types,
- eliminate all candidates that do not accept the number of actual parameters in the

- application,
- c. eliminate all procedure candidates in expression contexts, and all non procedure candidates in statement contexts,
- d. eliminate every candidate with a formal parameter type that is not a type of the corresponding actual parameter value, unless that would eliminate all candidates,
- e. if no candidate is valid, apply the implicit conversion rules of Section 2.4.2 to obtain a valid for reinterpreted actual parameters,
- f. if no candidates remain, an error is reported and nil is returned,
- g. if multiple candidates remain but would produce different values if applied to the actual parameters, there is an inconsistency in the program, an error is reported, and nil is returned
- h. otherwise one of the remaining choices is selected by the interpreter (because the choice does not affect the result, the choice will be either the most efficient or arbitrary),
- i. the body of the only remaining candidate is then instantiated with the actual parameter values as the initial values of the formal parameter attributes,
- j. if body is an operation (i.e. has statements) the body is evaluated and the resulting value returned,
- k. if the body is a type (i.e. does not have statements), a subtype with the parameters instantiated is returned.

#### 2.4.2. Implicit Conversions

# implicit value conversions

An actual parameter need not always be an example of the corresponding formal parameter type. This section describes the exceptions. These rules are applied only at the times and places described in Section 2.4.1. There may be multiple candidates for the actual parameter. The rules are applied only when none of the candidates satisfy the corresponding formal parameter type. All but the last rule is to be independently applied to each of the actual parameter candidates:

- a. if an example is used where a singleton type of which that example is the only example is allowed, replace the argument by the singleton type,
- b. if a singleton type is used where its only example is allowed, replace the argument by its only example,
- c. if a value of type  $t$  is used where a length one list or array of type  $t$  is allowed, replace the argument by a list or array containing only the argument,
- d. if a list or array of length one is used where a that one element is allowed, replace the argument by its only element,
- e. if a  $\text{lambda}([ ], t)$  is used where a  $t$  is allowed, apply the lambda,
- f. if none of these rules apply eliminate the candidate,
- g. if the same rule above will convert all the the actual parameter candidates, return the converted candidate, otherwise report an error and return nil.

#### 2.4.3. ““( $t$ : type, $x$ : exp $t$ ): $t$ is $x$ ;

# type qualification

Type qualification is used to explicitly specify the type of an expression. It can be used for disambiguation, emphasis or clarity, or to override the rules of section 2.4.2. Type quantification can be used to differentiate among definitions of constants, enumeration elements and types of the same name. When an actual parameter is type qualified, the  $t$  parameter replaces the formal parameter type when the rules of Section 2.4.2 are applied, and any further application of the 2.4.2 rules to satisfy the parent's formal parameter type are inhibited. There are also some special case rules as follows:

- a. if reference to a name is type qualified to “token”, the token itself is returned,
- b. if an expression is type qualified to “expression”, the expression itself is returned,
- c. if an aggregate is type qualified to t, t must be a composite type compatible with the aggregate,
- d. if an expression is type qualified to the universal type, any candidate elimination based on return type is inhibited, but the conversions of 2.4.2 are reenabled for the entire candidate list if needed.

## 2.5. Adjectives

Syntactically an adjective is an identifier that names a property and is used to modify a type to produce an anonymous subtype of that type. Adjectives are generally defined by authors and not built into the language. Adjectives generally return a subtype of their argument. In actual usage, a sequence of adjectives would normally be preceded by a quantifier to select an example from the type and followed by an improper noun naming a type.

2.5.1. adjective: type is lambda( [ type], type); #. adjective

Any lambda type or operation with one type valued formal parameter and a meta type as its return type may be used as an adjective. Lambda application as defined in Section 2.4 applies to adjectives.

2.5.2. “apply”( enumeration, type): type; #. enum apply

The first argument to apply may be an example of an enumeration type, but only if it is applied to a type that has a unique uninitialized attribute for which the enumeration value is an allowed value. The result of the application will be the subtype with the attribute initialized to the enumeration value.

2.5.3. “apply”( type, type type): type; #. meta type

When the second argument to an application is the type or all types, then a type may be used as the first argument to indicate the type of all subtypes of the first parameter. Thus type type is the meta type of all types, that is the type of which all types are examples. Enumeration type is the meta type of all enumeration types, that is the type of which every enumeration type is an example.

## 3. Ordered and Scalar Types

### 3.1. Ordered Types

#### 3.1.0. Syntax

op ::= "<" | ">" | "<=" | ">=" | ".." # relational operators

3.1.1. ordered: type; #. ordered types

An ordered type is any type whose examples are partially or completely ordered.

3.1.2. “<”( ordered, ordered): boolean; #. less than

Authors may define the less than operation for any ordered type. Less than must be transitive and non reflexive, but need not define a complete ordering. Less than returns true iff its first argument appears earlier in the (partial) ordering than the second argument, false if the second argument is earlier than the first and nil if the



arguments are unordered. The built-in less than operation applies to enumeration types and to numbers.

3.1.3. ">" (x: ordered, y: ordered): boolean is  $y < x$ ; #. greater than

The greater than operation is automatically defined whenever less than is defined.

3.1.4. "<=" (x: ordered, y: ordered): boolean is  $!(y < x)$ ; #. less or equal

The less than or equal operation is automatically defined whenever less than is defined.

3.1.5. ">=" (x: ordered, y: ordered): boolean is  $!(x < y)$ ; #. greater or equal

The greater than or equal operation is automatically defined whenever less than is defined.

## 3.2. Scalar Types

Any type whose examples are fully ordered is a scalar type. Scalar types include enumerations (section 3.3), machine integers (section 3.8), numbers (section XXX), integers and real numbers.. Integers and real numbers are not supported. Complex numbers are not scalars.

3.2.1. scalar: type;

#. scalar types

A scalar type is an ordered type whose values form a fully ordered sequence.

3.2.2. min( x: enumeration, type\_of x... ): type\_of x; #. enum min

The minimum operation applies to one or more arguments of the same enumeration type. It returns the minimum value among its arguments as defined by their ordering.

3.2.3. max( x: enumeration, type\_of x... ): type\_of x; #. enum max

The maximum operation applies to one or more arguments of the same enumeration type. It returns the maximum value among its arguments as defined by their ordering.

## 3.3. Enumeration Types

An enumeration type is a scalar type with a finite number of examples. The enumeration type constructor operation (section 3.3.1) is used to define an enumeration type by listing the names of its examples in order. Both characters (Section 2.2.3) and machine integers (see Section XXX) are examples of enumeration types.

3.3.1. enumeration: scalar type;

#. union type of all enumerations

3.3.2. enum( exp identifier... ): scalar type;

#. enum def

The enum operation constructs a scalar type by enumerating the names of its examples in order. Scalar types are described in Section 3.2. A side effect of a call on enum is to define those identifiers with their corresponding values throughout the scope of the current simulation.

3.3.3. "..."( exp identifier... ): identifier;



# unbounded enumeration

The presence of "..." after the last identifier in an enum definition specifies that the enumeration are also treated as a cycle with first immediately following last as might be the case with the days of the week. This specification affects the successor, predecessor, and enum ranges operations but does not affect the relational operations.

3.3.4. first( enumeration type): the type; #. first of an enumeration type  
First returns the least value of the enumeration type.

3.3.5. last( enumeration type): the type; #. last of an enumeration type  
Last returns the greatest value of the enumeration type.

3.3.6. "+"( x: enumeration, int): type\_of x; #. successor  
This operation returns the nth successor of its first argument. If the enumeration is cyclic the successor is modulo the number of elements in the enumeration. If it is not cyclic and the result is not in range, nil is returned. The int argument may be negative.

3.3.7. "-"( x: enumeration, int): type\_of x; #. predecessor  
This operation returns the nth predecessor of its first argument. If the enumeration is cyclic the result is modulo the number of elements in the enumeration. If it is not cyclic and the result is not in range, nil is returned. The int argument may be negative.

3.3.8. "-"( x: enumeration, type\_of x): int; #. enum difference  
This operation returns the number of elements from its first to second arguments in the order of the enumeration. If the enumeration is cyclic the result will be non negative, If it is non cyclic the result will be positive or negative depending on which argument is less.

3.3.9. ".."( x: enumeration, type\_of x): (type\_of x) type; #. enum range  
The operation returns a range of an enumeration type. If the enumeration is non cyclic and the first argument is greater than the second the range will be empty.

3.3.10. random( x: enumeration, #. enum random  
type\_of x): type\_of x; #.  
This routine returns a random value from a range of an enumeration type. Recall however that int is an enumeration type (see section 3.8.1).

### 3.4. Machine Integers

Machine integers are a built-in enumeration type for the integers in the range -32768.. 32767. Machine integers all have literals as defined in section XXX. Relational operations (see section XXX), enumeration operations (see section XXX) and the operations of this section apply to machine integers. Machine integers may be used anywhere numbers are required. Numbers that are integers in the range of machine integers may be used anywhere machine integers are required. It is illegal to use any other number where an integer is required. The bit by bit operations of sections 3.4.3. 3.4.7 operate on

3.4.1. int: enumeration type; -- integer in  $-2^{15}.. 2^{15}-1$  #. machine integers  
property int << number;

3.4.2. "\*" ( int, int): int; #. int multiply  
The integer product operation returns the least significant 32 bits of the product of its arguments when represented in twos-complement form.

3.4.3. permute( n: int): array( n, int); #. random permutation



This routine returns a random permutation of the integers 0.. n-1. The resulting values may be used as indices on any array of n elements to randomly permute the order of reference.

[3.4.4.]	band( int, int): int;	#. bit and
[3.4.5.]	bor( int, int): int;	#. bit inclusive or
[3.4.6.]	bxor( int, int): int;	#. bit exclusive or
[3.4.7.]	bnot( int): int;	#. bit complement
[3.4.8.]	bsr( x: int, n: int): int;	#. bit shift right
[3.4.9.]	bsc( x: int, n: int): int;	#. bit shift circular
[3.4.10.]	bsa( x: int, n: int): int;	#. bit shift arithmetic

### 3.5. Power Sets

A power set may be defined over any enumeration type. Set are immutable. They can be constructed as an aggregate of the members.

[3.5.1.]	set( t: enumeration type): immutable type;	#. power set type
[3.5.2.]	member( x:t, x: set): boolean;	#. set membership
[3.5.3.]	"*" ( set, set): set;	#. set intersect
[3.5.4.]	"+" ( set, set): set;	#. set union
[3.5.5.]	"-" ( set, set): set;	#. set difference
[3.5.6.]	"-" ( set): set;	#. set complement

## 4. Numbers

Numbers are imprecise real values. Numbers are also fully ordered scalar values. Special lexical syntax is provided for numbers (section A.1.3) and certain frequently constants are defined (section 4.1). Numeric operations include basic operation (section 4.2), integer conversion operation (section 4.3), transcendental and hyperbolic functions (Section 4.4), statistical functions (Section 4.5), and other numeric algorithm (Section 4.6).

### 4.1. Floating Point Numbers

#### 4.1.1. number( first: number, last: number): scalar type; #. numeric types

Each numeric subtype has a range first.. last. If not otherwise specified, first and last are respectively -infinity and infinity. Numbers are implemented in 32 bit IEEE floating point representation. This representation is precise to one part in  $2^{24}$  and has an epsilon of 1.19209209e-7, minimum positive value of 1.17549435e-38 and maximum positive value of 3.40282347e38. IEEE floating point also has special values for zero, infinity, -infinity and not\_a\_number.

#### 4.1.2. infinity: number; #. infinity

Infinity is a pseudo number used in IEEE floating point to represent a value in excess of those otherwise representable. Computations may be done on infinity. Negative infinity as -infinity is also supported.

#### 4.1.3. not\_a\_number: number; #. not a number

Not\_a\_number is a pseudo numeric value used in IEEE floating point to indicate erroneous or non representable computational results. Many of the operations of this chapter can generate not\_a\_number results and most will if given a not\_a\_number argument. When nil is used as an argument to a numeric operation it is converted to

not\_a\_number.

4.1.4. e: number is 2.7182818; #. base of natural logarithms  
A few useful numeric literal value are defined.

4.1.5. pi: number is 3.14159265; #. pi

## 4.2. Basic Numeric Operations

### 4.2.0. Syntax

exp ::= exp op exp   op exp   exp "..."	# in-fix and prefix expressions
op ::= "+"   "-"	# adding operators
"*"   "/"	# multiplying operators
"^"	# exponentiation operators

4.2.1. "+"( number, number): number;	#. add
4.2.2. "-"( number, number): number;	#. subtract
4.2.3. "*" ( number, number): number;	#. multiply
4.2.4. "/" ( number, number): number;	#. divide
4.2.5. "^" ( number, number): number;	#. exponentiation

4.2.6. "+"( number): number & it >= 0;	#. absolute value
4.2.7. "-"( number): number;	#. negate
4.2.8. "*" ( number): -1   0   1;	#. sign

4.2.9. min( number... ): number;	#. minimum
4.2.10. max( number... ): number;	#. maximum
4.2.11. sqrt( number): number;	#. square root
4.2.12. ln( number): number;	#. natural logarithm
4.2.13. log( x: number): number is (ln x)/ln 10;	#. logarithm base ten

## 4.3. Integer Conversion Operations

Integer conversion operations return integral values, i.e. real numbers that have zero as the fractional value. Because the range of the numeric representation is imprecise, integral values are precise only in the range -16777216.. 16777215.

4.3.1. floor( number): number;	#. floor
Floor returns the largest integer less than or equal to its argument.	

4.3.2. round( x: number, y: number): int24 is y *floor (x/y+0.5*y);	#. round
round( x: number): int24 is floor (x+0.5);	#. round to integer
Round returns the multiple of y which is nearest to x with value exactly between multiples of y rounded down. The second form of round returns nearest integer to its argument with 0.5 values rounded down. The second from is the same as the first with y = 1.	

4.3.3. trunc( x: number): int24 is *x*floor abs x;	#. truncate to integer
Truncate returns the value of its argument without its fractional value. Truncate reduces positive values and increases negative values.	

4.3.4. `div( x: number, y: number): int24` is floor  $x/y$ ; #. integer division

Integer division returns the number of integral times y divides x.

4.3.5. `mod( x: number, y: number): number` is  $x - y * (x \text{ div } y)$ ; #. modulo

Mod returns the remainder from integer divide. The result is always non negative.

#### 4.4. Trigonometric and Hyperbolic Functions

Trigonometric hyperbolic functions convert from directions to ratios of distances which are unitless. Arc functions convert from ratios of distances to directions. Although there exist implementations trigonometric and hyperbolic functions without units, each function has many implementations depending on whether its parameter cycles at 360, 2 pi, or some other value. The functions described below will accept any unit of direction.

4.4.1. `sin( direction): number;`

#. sine

4.4.2. `cos( direction): number;`

#. cosine

4.4.3. `tan( direction): number;`

#. tangent

4.4.4. `asin( number): direction;`

#. arcsine

4.4.5. `acos( number): direction;`

#. arccosine

4.4.6. `atan( distance, distance): direction;`

#. arctangent

4.4.7. `sinh( direction): number;`

#. hyperbolic sine

4.4.8. `cosh( direction): number;`

#. hyperbolic cosine

4.4.9. `tanh( direction): number;`

#. hyperbolic tangent

4.4.10. `asinh( number): direction;`

#. hyperbolic arcsine

4.4.11. `acosh( number): direction;`

#. hyperbolic arccosine

4.4.12. `atanh( number): direction;`

#. hyperbolic arctangent

#### 4.5. Statistical Functions

Statistical calculations and simulations require random number generation with a variety of distributions. All of the random number functions of this section have the general form:

`random( distributions, any type... ): number;` #. statistical random

where distribution is one of the random distributions of section 4.5.1 and the remaining arguments conform to the requirements of the particular distribution as described in sections 4.5.2.. 4.5.15. Fourteen random distributions are provided. The seed for the statistical functions is a system attribute (see section 10.1.XXX) that may be referenced or assigned by the program.

4.5.1. `distributions: enum( beta, binomial,`

#. random distributions

`chi_square, exponential, f_distribution, gamma,` #.

`lognormal, normal, poisson, triangular,` #.

`uniform, weibull, continuous, discrete);` #.

4.5.2. `random( beta, a: float, b: float); float;`

# beta distribution

This function returns a random value from a beta distribution where the density of the beta function is  $f(x) = x^{a-1} * (1-x)^{b-1} / \beta(a,b)$  for  $0 < x < 1$ , and where  $\beta$  is the

complete beta integral, random( chisq, df).

4.5.3. random( binomial, n: int, p: float): float; # binomial distribution

This routine returns a random value from a binomial distribution with n trials and a probability p of each trial. The result will be integral.

4.5.4. random( chi\_square, df: float): float; # chi squared distribution

This function returns a random value from a chi square distribution where df is the degrees of freedom.

4.5.5. random( exponential, mean: float): float; # exponential distribution

This function returns a random value from an exponential distribution with the given mean.

4.5.6. random( f\_distribution, dfn: float, dfd: float): float; # f distribution

This function returns a random value from an f (i.e., variance ratio) distribution where dfn is the degrees of freedom in the numerator and dfd is the degrees of freedom in the denominator.

4.5.7. random( gamma, a: float, r: float): float; # gamma distribution

This function returns a random value from a gamma distribution whose density is  $f(x) = ((a^r)/\text{gamma } r) * x^{(r-1)} * e^{(-a*x)}$ .

4.5.8. random( lognormal, scale: float, shape: float): float; # lognormal distribution

This function returns a random value from a lognormal distribution with the given scale and shape. If x has distribution random( lognormal, scale, shape), then ln x has distribution random( normal, scale, shape).

4.5.9. random( normal, mean: float, std\_deviation: float): float; # normal distribution

This function returns a random value from a normal distribution with the given mean and standard distribution.

4.5.10. random( poisson, mean: float): float; # poisson distribution

This routine returns a random value from a poisson distribution with the given mean. The result will be integral.

4.5.11. random( triangular, min: float, mode: float, max: float): float; # triangular

This function returns a random value of triangular distribution with given minimum, mode and maximum.

4.5.12. random( uniform, low: float, high: float): float; # uniform distribution

This function returns a random value from a uniform distribution between low and high. More precisely, the result may equal low but not high.

4.5.13. random( weibull, shape: float, scale: float): float; # weibull distribution

This function returns a random value from a Weibull distribution with the given shape and scale.

4.5.14. random( continuous, cp: array float, x: array float): float; # continuous

This function returns a random value from a continuous distribution by linear

interpolation from a user specified cumulative distribution. The array x specifies the x values for the cumulative distribution over the domain 0 to 1. The array cp specifies the corresponding cumulative distributions.

4.5.15. `random( discrete, cp: array float, x: array float): float; # discrete`

This function returns a random value from a discrete distribution over a user specified cumulative distribution. The array x specifies the x values for the cumulative distribution where the last value must be one. The array cp specifies the corresponding cumulative distributions.

## 5. Attributes

An attribute is any property or characteristic that is shared by every example of a type, but whose value may vary from example to example. Examples of attribute values and their corresponding types are green color, decisive attitude, 7 meter length, and vague idea. Attributes include the variables, record fields, array components, and list elements of object oriented languages.

### 5.1. Attributes Definitions

#### 5.1.0. Syntax

`prop ::= id ":" type ":" exp` # attribute definition

#### 5.1.1. `"attr_def"( identifier. type, the type): property; #. attribute definition`

An attribute definition specifies the name of the attribute, the type over which its value may range and the initial value of the attribute when constructing an example. The same attribute may be multiply defined either within a given define body or through inheritance of properties from supertypes. In such cases, the type of the attribute is the specialization (i.e. union of properties) of the multiple specifications. Two attributes of the same name may not be defined in the same local scope. If the name of an attribute is unspecified (see section 1.4.1), the attribute is anonymous and can be referenced only by dot quantification (see section 5.6).

#### 5.1.2. `"attr_def"( enumeration type, type, the type): property;` # indexable attribute

An `attr_def` defines multiple attributes of the same type and initial value. The names of indexable attributes must be elements of an enumeration type or subtype including subtypes of int. Indexable attributes with int names cannot be referenced or assigned without dot qualification or dot quantification.

#### 5.1.3. Attribute Initialization # attribute initialization

Each attribute definition specifies an initial value for the attribute. The value is specified as an expression that is interpreted once per instantiation of the attributes most local enclosing body. The unspecified indicator "?" (see section 1.4.1) may be used as the initial value in any attribute definition to mean that the initial value is not being specified at that point. The initial value specified in any local `attr_def` overrides any inherited initial value inherited from a supertype for the same attribute. If the initial value is unspecified locally, the specified values from all supertype specifications of the attribute will be used. If there are multiple initial value specification either from multiple local specification or when there is no local specification from multiple supertype specifications, all specification must have the same value or an error will be

reported. If no initial value is specified for an attribute, it is illegal to reference that attribute before it is assigned. If in the body of an operation, a reference to an attribute appears before the attribute is initialized or assigned, the reference will generate an uninitialized attribute error. In the case of initialization the error will be at compile time, and for assignment generally at run time.

#### 5.1.4. Formal Parameter Attributes

A formal parameter is a special form of attribute that is specified as part of the lambda definition (see Section 1.2.2) and initialized by application of the lambda (see Sections 2.4.1 and 1.2.5). Formal parameters are attributes of both the lambda subtype and of all its examples. Formal parameters are constant unless explicitly marked as variable. They are non private but never assignable from outside the body of the define.

#### 5.1.5. Scope of Visibility for Attributes

An attribute is visible only within the most local enclosing scope of its definition. That scope may be either a body scope or a compound statement. An attribute is not visible within an embedded scope of the scope of its definition if the embedded scope is the body of a define or contains an attribute definition of the same name. Formal parameters are exceptional in that they are visible not only within the formal parameter list but in the body of the lambda definition.

#### 5.1.6. `const( property): property;` $\beta$ #. constant specification

Const is a modifier for attribute definitions including formal parameter specifications. It specifies that the value of the attribute is not assignable from outside an example of the type to which it is a parameter. The absence of a const specification does not necessarily mean that the attribute is externally unassignable. Neither do a const specification prevent assignment to a attribute from inside an example.

#### 5.1.7. `var( property): property;` $\beta$ #. variable specification

Var is a modifier for attribute definitions including formal parameter specifications. It specifies that the value of the attribute is assignable from outside an example of the type to which it is a parameter. The absence of a var specification does not necessarily mean that the attribute is externally assignable.

#### 5.1.8. `always( a: exp, e: exp a.t): property;` $\beta$ #. always specification

The first argument to always must be a reference to an attribute. The second parameter is an expression of the same type as the attribute. Always specifies that the value of the attribute is always the value of the expression. That is, there is a mutual dependency among a and the attributes of e. If any of them are assigned, the other others are then updated without changing the assigned value. Inconsistent specifications that could cause an infinite loop are illegal. Attributes specified as always are called dependent attributes and are a form of continuously evaluating expression.

### 5.2. Operations on Attributes

#### 5.2.0. Syntax

<code>exp ::= id</code>	# attribute reference
<code>stat ::= exp "!=" exp</code>	# assignment statement

#### 5.2.1. `"ref"( exp): any type;` #. attribute reference

The attribute reference operation returns the current value of the named attribute. The argument must be an identifier naming an attribute that is visible at the point of the reference. If the attribute has not been initialized, an error will be reported and the nil value returned. The visibility rules guarantee that attributes and defines of the same name are never simultaneously visible. Syntactically, the appearance of the identifier name alone constitutes a reference.

5.2.2. “:=” ( exp, any type): action; #. attribute assignment

The attribute assignment operation replaces the value of an attribute by the value of its second parameter. The attribute must be variable, must have an identifier name and must be visible at the point of the assignment.

5.2.3. type\_of( exp): type; #. type of attribute

The argument to type of must be a reference to an attribute or an element of an enumeration type. It returns the attribute’s type as specified in the att\_def or the enumeration type as specified by an enum constructor.

5.2.4. has( any type, exp): boolean;  $\beta$  # attribute predicate

The attribute predicate, “has”, returns true iff the type has an attribute referenced by the expression. This operation is rarely, if ever, needed because in most cases it is equivalent to comparing the value of the expression to nil.

5.2.5. is\_unspecified( exp): boolean;  $\beta$  # unspecified predicate

The unspecified predicate returns true iff the value of the attribute is unspecified.

5.2.6. “attr\_init”( any type... ): any type; #. multiple attribute init

This is a special internal operation that is called implicitly when there are multiple potentially conflicting specifications of the initial value of an attribute, and the translator is unable to prove that they will produce the same value. Each value is passed as an argument to attr\_init and the attribute initialized the return value of attr\_init. Attr\_init compares its arguments for equality. If they are all the same it returns one of them. Otherwise, it reports an error and returns nil.

## 5.3. Composite Types

### 5.3.0. Syntax

exp ::= “[{ exp “,” } exp ] ”	# aggregate constructor
exp “.” id   exp “.” int_lit	# dot qualified reference
exp “.” “( exp )”	# dot quantified reference
exp “.” “[ exp ]”	# indexed reference

5.3.1. composite: type; #. composite type

A composite type is any type that has attributes, formal parameters, indexed attributes and external attributes. All records, arrays, lists and parameterized types are composite.

5.3.2. immutable: composite type; #. immutable type

An immutable type is any type whose examples do not have any attributes whose values change over time.



5.3.3. “[ ]”( any type... ): composite;  $\beta$  #. aggregate

The aggregate constructor returns a new composite value with one attribute for each actual parameter. Each attribute is initialized to the corresponding actual parameter value. Aggregates may be used to construct mutable or immutable values. The potential syntactic ambiguity between indexing and application to aggregates is always resolved in favor of indexing. If application to an aggregate is desired, the aggregate must be parenthesized.

5.3.4. “.”( composite, identifier | enum\_lit): any type; #. dot qualified reference

Dot qualification returns the value of an attribute or formal parameter given a composite value and an identifier or enum literal. The second parameter must be the name of an attribute, definition or formal parameter defined in the body of the composite's type. Dot qualification can also be used to reference the value of a parameter of a subtype of a parameterized type. If the composite does not have a local attribute of the given name but instead has a composite attribute with an attribute of the given name, the value of the indirect attribute will be returned. This latter process may be applied recursively.

5.3.5. “index”( composite, exp): any type; #. indexed reference

The indexed reference operation references the value of an attribute given a composite having that attribute and an enumeration value specifying which attribute is intended. The attribute must have been defined as an indexable attribute. The dot qualification operation of section XXX may be used when the expression is an enumeration literal. The potential syntactic ambiguity between indexing and application to aggregates is resolved in favor of indexing. If application to an aggregate is desired, the aggregate must be parenthesized.

5.3.6. “dot”( composite, exp): any type;  $\beta$  #. dot quantified reference

The dot quantified reference operation references the value of an attribute given a composite having that attribute and a quantified expression specifying which attribute is intended. The second argument is evaluated in the context of the body of the first argument, and the resulting value is returned.

## 5.4. Mutable Types

5.4.1. mutable: composite type; #. mutable type

A mutable type is any type whose examples have an identity separate and apart from the values of their attributes. Mutable types may be distinguished by having state, by having assignable attributes, and by the need for an explicit operation, new, to create a reference to an example.

5.4.2. “.”( mutable, identifier | enum\_lit, any): action; #. dot qualified assignment

Dot qualified assignment is analogous to dot qualified reference except that the first argument must be a mutable and the value of the attribute is replaced rather than referenced. The third argument specifies the new value.

5.4.3. new( mutable type): the type; #. create reference

New creates a reference to an example of a mutable type. Each new reference is unique and distinguishable from all other references. Multiple references however may



refer to the same example.

5.4.4. "index" (mutable, exp, any type): action; #. indexed assignment

The indexed assignment operation assigns the value of an attribute given a mutable value having that attribute, an enumeration value specifying which attribute is intended, and a value to be assigned. The attribute must have been defined as an indexable attribute. The value must be of the attribute's type. The dot qualified assignment operation of section XXX may be used when the expression is an enumeration literal.

5.4.5. "dot" (mutable, exp, any type): action;  $\beta$  #. dot quantified assignment

The dot quantified assignment operation assigns the value of an attribute given a mutable value having that attribute, a quantified expression specifying which attribute is intended, and a value to be assigned. Dot quantified assignment assigns the value to the attribute that would be referenced by the second argument if it were evaluated in the context of the first argument. The value must be of the attribute's type.

## 5.5. Sequences and Arrays

5.5.1. sequence( t: type): composite type is { length: int := 0; }; #. sequence type

A sequence is a composite and indexable value with a length attribute. The index and dot qualified reference operations of Section 5.3 apply to sequences. If the sequence is mutable, the index and dot qualified assignment operations of Section 5.4 also apply. The length is the number of indexable attributes and t is the type of each indexable attribute.

5.5.2. array( first: enumeration, last: type\_of first, #. array type  
t: type): sequence type; #.

An array is a fixed length immutable sequence. The values of first and last do not change throughout the life of an array. Length is always last-first+1.

5.5.3. catenate( sequence... ): array;  $\beta$  #. array catenation

The catenate operation returns an array of the all the elements of its argument in the order given. The length of the result is the sum of the length of its arguments. The indices of the return value are 0.. length-1. Any argument that is not a sequence is interpreted as a one element list, with its value as the only element (see Section 2.4.2.c).

5.5.4. get( sequence, first: enumeration,  $\beta$  #. get subarray  
last: type\_of first): array; #.

The get subarray operation returns an array that has the same values as the first.. last elements of the sequence. The indices of the return value are 0.. last-first.

5.5.5. character: type is enum( "\00" ... '\FF"); #. character type

Characters are an enumeration type consisting of all ascii characters in their numeric order. This may later be changed to Latin one or a 16 bit character code.

5.5.6. string: type is sequence character; #. character string type  
A string is a sequence of characters.

## 5.6. Lists

5.6.1. `list( t: type): mutable sequence type;` `#. list type`

A list is a variable length mutable sequence. The elements of a list are assignable and are indexed by the integers 0.. length-1. Elements may inserted or removed from a list at any index and the length is adjusted accordingly.

5.6.2. `pop( list): (the list).t;` `#. pop list`

Pop returns the value of the last element of the list, and reduces the length of the list by one. If the list is viewed as a stack or last-in-first-out queue, then this is the stack pop operation. This is a constant time operation.

5.6.3. `append( x: list, sequence x.t): action;` `#. append to list`

Append inserts each item of the sequence at the end of the list. The sequence's items must be of the type required by the list.

5.6.4. `push( x: list, x.t... ): action;` `#. push last-in-first-out`

Push inserts each of its arguments, beyond the first, at the end of the list. The last element, i.e., `x[ length(x)-1]`, . If the list is viewed as a stack or last-in-first-out queue, this operation is the stack push operation. This is a constant time operation.

5.6.5. `truncate( x: list, n: int): action;` `#. truncate list`

5.6.6. `insert( x: list, int, x.t...): action;` `#. insert into list`

5.6.7. `remove( list, int type): action;` `#. remove item from list`

5.6.8. `fifo_push( x: list, x.t... ): action;` `#. push first-in-first-out`

Fifo push inserts each of its arguments, beyond the first, at the beginning of the list. The first element, i.e. `x.0`, of the resulting list will be the last argument. The time for this operation is proportional to the length of the list. If lower cost fifo queueing is required the tree sort 3 algorithm of Section 4.6.2. should be used.

### [5.7.] External Attributes

An external attribute is an attribute of a mutable type that is defined and managed separate external to examples of the type. The values of the external attributes for all existing references to examples of the type form a single example of type attribute.

[5.7.1.] `attribute( mt: mutable type,` `#. external attribute`  
`t: type, t): mutable type;` `#.`

The first argument to the external attribute type is the type to which the attribute applies. The second argument is the type of values of the attribute, and the third argument is the initial value of each attribute.

[5.7.2.] `"dot"( a.mt, a: attribute): a.t;` `#. external attribute reference`

This operation returns the current value of external attribute `a` of value `x`. The value `x` must be a reference to an example of the mutable type of the external attribute. The reference will be valid whether the attribute was created before or after the creation of the example `x`.

[5.7.3.] "dot" ( a.mt, a: attribute, a.t): action; #. external attribute assignment

This operation assigns the value of external attribute a of example x. The value x must be a reference to an example of the mutable type of the external attribute and the third argument must be an example of the attribute value type of a.

## 6. Operations and Statements

### 6.1. Defining Operations

An operation is a type describing a relationship among the types of its formal parameters and its return type. An operation may be evaluated or executed to produce a value or state change effect given appropriate actual parameter values. Each such evaluation is an example of the operation type. Operations are a supertype of functions, procedures, actors and simulations.

#### 6.1.0. Syntax

prop ::= name [ fpl ] ":" type [ "is" body ]	# lambda definition
name [ fpl ] "is" body	# define with unspecified type
name ::= id   "\"q" op "\"q"	# name: id or quoted operator
fpl ::= "(" [ [ { fp "," } ] fp [ "..."] ")"	# formal parameter list
fp ::= [ id ":" ] type	# formal parameter
stat ::= exp [ "(" ")" ]	# procedure call - no args
exp exp	# procedure call - one arg
exp "(" exp { "," exp } ")"	# procedure call - multiple args
"?"	# unspecified action

6.1.1. "define" ( token, t: lambda type, # define an operation  
exp t.rt): property;

The definition of a operation is a modified form of the define operation as given in section 2.1.1. It differs in that the second parameter in the definition of an operation must be a lambda type and the body must include statements. The body of an operation is a description of a class of computations. Each application of an operation creates one example of the class as a evaluation instance of the computation. The return type of the lambda must be a type of the value resulting from evaluation of the body or action is no value results. If the body of an operation is unspecified (see section 1.4.1), it indicates only that the definition is incomplete as specified at this point. Syntactically, the fpl, result type or body may be omitted. If the fpl is omitted, it is taken as a list of length zero. If the type is omitted, it is taken as unspecified which is equivalent to all | action. If the body is omitted it is either unspecified or specified elsewhere. Every call on define must have an explicit type or body, although "?" may be used in either case.

6.1.2. function: lambda type; #. side-effect free functions

Function is the type of all operations that are both side-effect free and return a value. The apply operation of Section 2.4.1 applies to function. The application of a function is called an expression.

6.1.3. Side-effects in Expressions # side-effects in expressions

Value returning operations may have side-effects. In general such effects should be rare and should not be visible at the points of call on the operation. Their most common use would be internal diagnostics and debugging such as counting the number of time

an operation is called. The language does not specify the order of multiple side-effects in an expression or statement. When order matters, call only one operation with side-effects per statement. The language does not guarantee that the nominal side-effects of value returning operations will be preserved over all optimizations.

6.1.4. action: type; #. type of all actions

Action is the pseudo type of the return value of procedures. Action is roughly equivalent to the void class of the C language.

6.1.5. procedure: lambda( ?, action) type; #. procedure type

Procedure is the type of all operations that do not return a value. The return type of a procedure is always action. Application of a procedure is called a statement.

6.1.6. "?": action;  $\beta$  #. unspecified action

An unspecified action may be used in a statement list to indicate that there may be additional actions at that point that are unspecified. In the absence of such a specification consecutive statements are assumed not to have any intervening actions within that actor.

6.1.7. return( any type): action; #. return value

Every execution sequence within the body of a function or other value returning operation must include a call on the return procedure of one argument. The actual parameter to return must be an example of the return type of the operation. The effect of return is to return its argument value to the point of call on the operation and terminate evaluation of the operation's body.

6.1.8. return(): action; # procedure return

Return permanently discontinues execution of the procedure and resumes execution of its caller just beyond the point of call. Every execution sequence within the body of a procedure must include a call on the parameterless return procedure. Return is however implicit at the syntactic end of each procedure body.

## 6.2. Sequential and Conditional Control

Control structures are evaluated for the side-effects they produce in the state of the system, and generally do not return values. Control structures are defined as procedures. Each application of a procedure is a statement. If a parameterless procedure is called where a statement is required, the actual parameter parentheses may be omitted without altering the interpretation.

### 6.2.0. Syntax

```
stat ::= str_lit           # statement label
      | "{" [{ prop ";" } ] "}" # compound statement
      | "if" exp "then" stat [else_part] # conditional statement
      | "select" exp of [{ "case" type "then" stat } ] [else_part] # select statement
else_part ::= ";" "else" stat
```

6.2.1. Statement Label # statement label

A statement label is any string literal that appears in the position of a statement. Statement labels are used to describe the purpose of the following statements and to

label that point in the statement sequence.

6.2.2. null( any type... ): action;

#. null procedure

The null procedure is the do nothing operation. It is generally called without arguments to explicitly indicate that the absence of a statement. Null is also a means of disposing of unwanted values in a statement context. In the latter form it is analogous to a C language function call in a statement context. In the special case of null new t for some type t, it is as if there were instead an attribute definition of the form ? : t := new t, at least for the purposes of quantified reference, thus allowing quantified reference to the new t.

Examples:

null;  
null f( 3, 8);  
null new project;  
begin the project;

6.2.3. "cpnd\_stat"( property... ): type;

# compound statement

A compound statement allows multiple statements to be combined into a single statement that will be evaluated sequentially in the left to right (top to bottom) order of their appearance. The statements of a compound statement may also include any number of declarations. Any define or attribute definition that appears within a compound statement is both local and private to that compound statement and cannot be referenced from outside. A compound statement defines a visibility scope that is exceptional in that attributes of its parent scope (that are not hidden by local attributes of same name) can be referenced within the compound statement.

6.2.4. "if"( boolean, exp action, exp action): action;

#. if conditional

The if conditional operation evaluates either its second or third actual parameter depending on whether the value of its first parameter is true or false respectively. If the third argument is not specified in the syntax, it is interpreted as null.

6.2.5. "select"( all, list [exp type, exp action],  
exp action): action;

#. select  
#.



The select operation evaluates the type fields of its second argument in order until it encounters one whose value is the type of its first argument. At that point it evaluates the corresponding action of the second argument and returns. If none of the types match, the third argument is evaluated. If the third parameter is not specified in the syntax it is interpreted as null.

6.2.6. goto( str\_lit): action;



#. goto operation

The goto statement allows explicit transfer of control to any labeled point (see section 6.2.1) within the body of the most local enclosing body of a definition providing the label is not within a more local body or a more local compound statement that contains attribute definitions. Use of this operation is recommended only where the high level control structures are inadequate for the application.

### 6.3. Iterative Control

stat ::= "for" [ id ":" ] exp "do" stat [else\_part]

# iterative statement

6.3.1. "for"( identifier, exp list | type, #. for loop  
exp action, exp action): action; #.

A for statement causes its third argument to be evaluated repeatedly the number of times indicated by its second argument. If the third argument is not evaluated at all, then the forth argument is evaluated once. The second argument must have an iterator, either each (section 6.3.2) or every (section 6.3.3), and also may have the reverse iterator (section 6.3.4). The for selects one value from the value set of its second argument at each iteration. For each value selected the body (i.e., third parameter of the for) is evaluated. When the second argument is a list, finite enumeration or mutable type, the first argument may be an identifier that names the selected value during execution of the third parameter, the order is that of the list or enumeration, and it may also be used to reference the selected value. When the second argument is an infinite type, the third parameter is repeatedly executed until explicitly terminated by an exit, return or terminate operation, and the selected value is not accessible.

6.3.2. each( list | type): type; #. each, the static iterator

Each is a quantifier that selects from a statically determined set of values specified by its argument. The argument may be a list, enumeration type, mutable type, numeric type or other type. For most purposes each is the same as the quantifier, any.

For lists, the set of values are the list's elements. The elements of list are determined once at the application of each and unchanged thereafter.

For mutable types, the set of values are the references to examples of type created by application of new. The set of references is determined once at the time of the application of each and is not changed thereafter.

For immutable types (including enumeration types, integers and numbers) the set of value are the examples of the type. The type is determined once at the application of each and is not changed thereafter.

The primary use of each is as the second argument to for (section X.X.X) where value from the list or type are selected one at a time in list, enumeration or other order.

6.3.3. every( exp list | type): type; #. every the dynamic iterator

Every is a quantifier that selects from a dynamically determined set of values specified by its argument. The argument may be a list, enumeration type, mutable type, numeric type or other type. For most purposes every is the same as the quantifier, all.


For lists, the set of values are the list's elements. The list is determined once at the application of every, may be shared and thus may change in length or content between selections.

For mutable types, the set of values are the references to examples of type created by application of new. The set of references is determined dynamically at each selection.

For enumeration types (including integers) the set of value are the examples of the enumeration. The first element of the enumeration is determined once at the application of every, but the last example is determined dynamically at each selection.

For numeric and other types, the set of values are the examples of the type and are determined dynamically at each selection.

The primary use of every is as the second argument to for (section X.X.X) where the set of values are selected one at a time in list, enumeration or other order.

6.3.4. reverse( type): type;  #. reverse iterative selection  
When the reverse iterator is applied to an each or every type, the values are



generated in reverse order.

6.3.5. `exit( exp): action;`  $\beta$  `#. exit from loop`

The exit operation may be executed anywhere within the body (i.e. third actual parameter) of a for statement to cause immediate transfer of control to the point immediately following the for statement. Exit's argument must be the same identifier as the first argument to the for. Exit may be used to exit any lexically enclosing for statement regardless of how many intervening control structures there may be.

6.3.6. `cycle( exp): action;`  $\beta$  `#. cycle loop`

The cycle operation may be executed anywhere lexically within the body (i.e. third actual parameter) of a for operation whose control attribute's name is the parameter to cycle. Cycle cause an immediate transfer of control to the end of the current iteration of the body of the for.

[6.3.7.] `reduce( f: lambda([f.rt, f.rt], ?), array f.rt): f.rt;` `#. binary reduction`

The reduction operation begins with the first element of the array and then successively applies the binary function f between the result and the next element of the array until all elements have been used. The final value is returned. Reduce is functionally equivalent to the `op/x` reduction operation of APL.

[6.3.8.] `expand( f: lambda([f.rt, f.rt], ?), array f.rt): list f.rt;` `#. binary expansion`

Expand performs the same computation as reduce, but returns an array of the first, all intermediate and final value of the computation. Expand is functionally equivalent to the `op\x` expansion operation of APL.

## 6.4. Interpretation Semantics

### 6.4.1. Implicit Assumptions

Easel allows incomplete and imprecise specifications, intensional references, and computations on quantified types. One consequence is that it is often impossible to complete a computation without obtaining additional information from outside the system or by making assumptions. It is in general impractical to interrogate the user in every case that additional information is needed. Often the user would not know, have ready access to, or be able to obtain accurate answers.

The easel interpreter instead makes assumptions in the following priority:

- assume a choice that does not lead to an inconsistency over one that does
- in a form `x = true` where the value of x is unknown, assume `x = false`
- in a form `x = false` where the value of x is unknown, assume `x = true`
- in any conditional control operation assume the else condition if the values of the precondition for the previous choices cannot be determined
- in uses of quantified expressions always make the worst case assumptions

### 6.4.2. Explicit Assumptions

In general, evaluation of programs requires continued maintenance of worst case assumptions for all quantified expressions, recording of all assumptions made by the interpreter (except those that are obvious because all alternatives lead immediately to an illegal execution), non deterministic execution (either by backtracking, sidetracking, or symbolic execution), and a theorem proving mechanism. Although the interpreter's capabilities will improve over time, it will never be able to prove all valid theorems.

More importantly, in the absence of complete and precise information, not all questions will be answerable nor all computations feasible from the available information. Instead the interpreter will make assumptions that enable the simulation to proceed, will add the assumptions to types of the current execution to ensure consistent assumptions, and will report the assumptions made to the user for analysis and later incorporation, either positively or negatively, into the program.

6.4.3. `assert( exp boolean)`: action; #. point assertion

A point assertion is a property of each instantiation of an active (i.e. functional, procedural, actor, or simulation) type. It asserts that some condition will be true at that point in the activity of each example of the type.

6.4.4. `failure`: action;  $\beta$  #. failure specification

A failure specification marks a point in the execution sequence which should not be reached. In any deterministic evaluation, executing a call of fail will cause the program to terminate with a error report. In a non deterministic program evaluation, as might arise from unresolved quantifiers such as "some", the interpreter may tentatively try alternative execution sequences depending on which candidates for "some" is involved. If some but not all such paths fail, no error is reported. Instead, the corresponding examples are eliminated as candidates. Fail should be called whenever an inconsistency is detected. Fail is called for all failure conditions in built-in procedures. Built-in functions often return nil instead.

6.5. Mobile Code and Expressions as Data  $\beta$

Security and survivability often require that computational descriptions be treated as data. Programs and parts of programs are computed in one node of a network, transferred to another node, and executed there. This ability is required to simulate down loadable code, java applets, viruses, Trojan horses, and other active content. Outside the domains of network security, mission survivability and infrastructure assurance, expression valued data may be used for a variety symbolic computations analysis and synthesis of computer programs, software transformations and optimizations, symbolic mathematics including algebra and calculus, and , program optimization, differentiation, integration and algebraic manipulations.

6.5.1. `translate( string | file)`: expression; #. translate

The translate operation takes a source text file or string and returns the expression represented by the string.

6.5.2. `token`: enumeration type; #. token type

6.5.3. `identifier`: token type; #. identifier type

Identifiers are a subtype of token.

6.5.4. `expression( t: type)`: type is  $\beta$  #. expression type  
(list expression) | any type; #.

The expression type is the type of any program component or mobile code. When an expression takes the form of a list of expressions it the first list item is interpreted as an expression for an operation and the remaining items as expressions for the actual parameters. Generally, any expression in a program will be evaluated, but by type qualifying it to the expression type (see section 2.4.3), the expression itself will be



passed as the argument. The evaluation of an actual parameter can also be inhibited by and exp specification (see section 6.5.5) on the corresponding formal parameter type.

6.5.5. `exp( t: type): type;` #. formal expression specification

The exp specification may be used only as a formal parameter type. It specifies that the actual parameter must be of type t, but that the formal parameter attribute type is expression t. When a formal parameter type is specified as exp, the actual parameter is not evaluated at the time of application and is instead interpreted as the value of the formal parameter attribute.

6.5.6. `eval( e: expression): e.t;`  $\beta$  #. eval

Eval is the interpreter for all expressions of the language. If the expression e was obtained by a qt operation, then eval returns the value of e interpreted in the local context of the call on eval. If e was obtained by inhibiting an actual parameter evaluation as specified by a formal parameter of type exp, then e is evaluated in the context of that call. In the latter case, eval will fail if evaluation context is not an existing frame of the current actor.

6.5.7. `eval( e: expression, x: composite): e.t;`  $\beta$  #. context eval

This evaluation operation may be used to evaluate a expression in a designated context. The context must be an example of a composite type.

## 7. Actors, Neighbors and Simulations

### 7.0. Syntax

`stat ::= "take" exp "to" stat;` # duration specification

### 7.1. Actors

7.1.1. `actor: mutable type;` #. autonomous participants

An actor is any autonomous participant of a program or simulation. The actors of a simulation are of equal status and are independent of their parents (i.e. an actor may outlive its ancestors). Props and actors may be used the control type of an iterative statement (see Section 6.3). Actors may have any number of author defined formal parameters and attributes. Each actor also has four private attributes:

`sim: simulation := ?;` # the scheduler controlling this actor

`fs: list float := [ ];` # floating point stack of this actor

`tot: time := infinite;` # the time out time for current operations of the actor

`next: actor := ?;` # the next actor in whatever queue the actor is in currently.

7.1.2. `new( actor type): the type;` # new actor

New creates a new actor. This operation is a special case of the new operation of Section 5.4.3. Each actor is created, its attributes initialized, a reference to the actor is returned to its parent, and only then are evaluation of its parent and of the remainder of its body continued in parallel. Actors are evaluated in parallel with each other in either real or simulated time, and with concurrent or interleaved semantics depending on their scheduling regime (see section XXX).

7.1.3. `pronoun self: actor;` #. self, the current actor pronoun

Any actor may reference itself using the pronoun self. Self always refers to the dynamically current actor.

7.1.4. `terminate()`: action;

#. terminate self

Terminate immediately and permanently discontinue execution of the current actor. Any existing references to the actor will however remain valid. Every execution sequence within the body of an actor must include a call on the parameterless terminate procedure. A call on terminate is however implicit at the syntactic end of each actor body.

7.1.5. `indivisible( exp action)`: action;



#. indivisible actions

Indivisible is a statement modifier. It specifies that its argument is an indivisible action. That is, all state changes caused by evaluation of the argument appear as a single state change from the perspective of all other actors.

7.1.6. `prop`: type is mutable & ! actor;

#. non autonomous participants

A prop is any mutable component of a program or simulation that cannot change its own or another components state. That is, any mutable example that is not an actor may be viewed as a prop. Props and actors may be used the control type of an iterative statement (see Section 6.3).

7.2. Neighbor Relationships



Neighbor operations are used to describe the relationship among actor and between actors and props. The neighbor operations of this section allow the relationships among actor to be described as operations that can be performed by one actor, the perpetrator, that change the state of another, the victim. By defining such operations within the victim, the can operation may be used to limit which actors may be perpetrators and the private specification to limit which attribute may be referenced. The effect is that there is no global visibility among actors except as explicitly allowed by actor descriptions.

7.2.0. Syntax

`prop ::= exp "can" body`

# neighbor relationship

`body ::= prop | "{" [{ prop "," } ] "}"`

# body of type definition

7.2.1. `"can"( exp actor type, property)`: property;

#. neighbor relationship

The can operation specifies that only actors of given type specified in the first argument can apply definitions defined in the second argument. The second argument must be either a single lambda define or a compound declaration of lambda definitions.

7.2.2. `"cpnd_decl"( property... )`: property;

#. compound declaration

A compound declaration may be used to combine multiple definitions for use in places where a single definition is normally expected. The compound declaration is most useful when specifying neighbor relationships in which several operations apply between the actors. A compound statement is syntactically the same as a "body" but has no effect on visibility. Every definition within a compound declaration has the visibility it would have if it were defined immediately outside the compound declaration.

7.2.3. `private( property)`: property;

#. private specification

A private specification is a modifier to an `attr_def` or local define. It specifies that the attribute or definition cannot be reference by any actor other than the most

enclosing actor of the attr\_def or define.

### 7.3. Simulations

A simulation defines a world of interactions among examples of many different types. It is a belief system about some real or imagined world. Types may be defined local to a simulation and must be self consistent and consistent with other types of that simulation, but need not be consistent with any enclosed, parallel or enclosing simulation. Each simulation constitutes the limit of the visibility scope for all types and operations defined within that simulation. Each simulation has its own simulated time regime, scheduler and simulation controls.

7.3.1. simulation: actor type; #. the simulation type

Each simulation is an actor of the program or parent simulation. The specified return time of a simulation must be "simulation". Actors within a simulation have concurrent semantics.

7.3.2. pronoun sim: simulation is { #. current simulation pronoun

Sim is a pronoun used to reference the most global scope within the current simulation. Both the built in simulation attributes of Sections 7.3.3 through 7.3.X and author defined attributes of a particular simulation type may be referenced by dot qualification on sim. They may also be referenced by name within the simulation. The following simulation attributes are built in. In addition, there are private built-in simulation attributes as described in section XXX.

7.3.3. clock: time := ?; #. simulated time

Clock is an attribute of every simulation. The value of clock is the amount of simulated time between the start of the simulation and the current simulated time. Outside of a simulation, clock has the same value as does system.rtc (see section 10.1.7).

7.3.4. granularity: time := 0 \*seconds; #. granularity of simulation

The granularity attribute is a simulation parameter that controls time precision of the simulation. Granularity is the minimum increment to the simulated clock. The granularity can affect the outcome of simulations because large grain simulation may hide fine grain time dependencies. Granularity is initialized to zero. This guarantees maximum precision, but in some case with increased computational costs.

7.3.5. speed: number := infinity; #. speed of simulation

The speed attribute is a simulation parameter that controls the real time speed of the simulation. Speed is an upper bound on the ratio between simulated and real time. Speed affects the depiction but not the functional results of a simulation. The speed is initialized to infinity which guarantees that the simulation will run as fast as possible within the processing capabilities of the system.

7.3.6. scale: enum( uniform, absolute, adaptive) #. scale of simulation  
:= uniform; #.

The scale attribute is a simulation parameter that controls variances in the simulation speed. Uniform uses the value of speed whenever possible. Absolute attempts to make up for computationally intensive intervals by increasing the speed during periods of less activity to maintain the absolute speed ration over extended periods. Adaptive increases the speed logarithmically during periods of inactivity in

the simulation.

#### 7.3.7. assumptions: list type | expression boolean := [ ]; #. assumptions

The assumptions attribute contains all semantic assumptions made by the interpreter in order to continue execution of a simulation. Any assumption added to this list becomes part of the belief system of the simulation. Users may also specify properties or make assertions that will be added to this list. The assumptions should be output for analysis at the end of the simulation. Some or all of the assumptions, or their contradictions, should then be made explicit in the source program.

#### 7.3.8. Private Attributes of Simulations

Each simulation has the following private attributes:

runq: actor := ?; #. circular fifo queue of currently executing actors  
timeq: actor := ?; #. circular fifo queue of actors doing wait time or take  
condq: actor := ?; #. circular fifo queue of actors currently waiting for condition  
eventq: actor := ?; #. circular fifo queue of actors currently waiting for an event  
# only suspended and terminated actors are not in a queue  
depictions: depiction := ?; }; #. depictions changed since last clock increment

#### 7.3.9. processor: actor type; #. processors

A processor is a special actor type that can be used to simulate computers with multiple threads of control. Processor have all of the properties of actors including concurrent execution with other actors of the simulation. Processors also have their own scheduling regime in which subactors can be created and scheduled. Actors of a processor however have interleaved semantics with the processor consuming the sum of all its thread times. Processor are not belief system and do not limit the scope of definitions.

### 7.4. Belief Systems

#### 7.4.1. visibility of simulations

Simulations define the visibility scopes of all global defines. That is, global defines made anywhere within a simulation are visible in a scope that is global to the body of the simulation, but local to the simulation's formal parameters. Global scopes of simulations do not contain attribute definitions. Each simulation scope hides all attributes and defines of all more global simulations, so that each simulation acts as an independent belief system. All system level definitions, including built-in language defines, however, are visible through out every simulation. Language defined attributes are visible in any scope in which it is not hidden by a attribute or define of the same name.

#### 7.4.2. hide( property): property; $\beta$ #. hide system level definition

The hide specification may be used to hide system level definitions throughout a current simulation. Each hide operation, hides all system level definitions of the same signature as its argument and also hides any system level attribute of the same name. The parameter to hide should be a define with complete signature but no body.

#### 7.4.3. parent( simulation): simulation; $\beta$ #. parent simulation scope

Parent returns the parent of a simulation. If the argument is the outermost simulation of the program then the program itself is returned. If the argument is the

program or the system then the system is returned.

## 7.5. Simulated Time

### 7.5.1. time: dimension;

#. the time dimension

Time is a primary dimension of any simulation or description of the real world. Time may be used to measure either duration or absolute time since the beginning of the simulation.

### 7.5.2. Units of Time

seconds: unit time;

#. seconds

minutes: unit time is 60 \*seconds;

#. minutes

hours: unit time is 60 \*minutes;

#. hours

days: unit time is 24 \*hours;

#. days

years: unit time is 365.2475 \*days;

#. years

Time may be measured in a variety of units including seconds, minutes, hours, days, and years. The unit conversions between days and years is an imprecise average and in general should be used only when dealing with large numbers of years. As with all units the value of one unit is arbitrary as long as it is used consistently. The implementation uses a time value of zero to represent midnight January 1, 2000 and one to represent a duration of one day. This combination gives time with 24 bit precision over the approximate range 1e-15 seconds to 1e15 years.

### 7.5.3. date(number...): time;



#. date compression

The date compression function takes a list of numbers specifying the year, month, day, hour, minute and second respectively. If the list has less than six elements the missing tail values are interpreted as zero. The parameter values are normally integers but may be fractional. This function uses precise values for the number of days in each month and year as prescribed by the Georgian calendar.

### 7.5.4. date(time): list number;



#. date expansion

The date expansion computes as list of the year, month, day, hour, minute and second from an absolute time. If local time is desired the time zone should be added before calling this function.

### 7.5.5. wait(interval: time): action;

#. wait

The wait operation specifies that the current actor will do nothing for the specified interval.

### 7.5.6. "wait until"(time): action;



#. wait until

The wait until operation specifies that the current actor will do nothing until the given time.

### 7.5.7. when(exp boolean): action;



#. when condition is true

The when operation specifies that the current actor will wait until the given condition is true. More precisely, the actor is suspended until the interpreter detects that the condition is true. If the value of the boolean changes from false to true and back to false between clock increments the true condition may or may not be detected. If the condition is initially true, the actor will not be suspended, but at the implementation level will be queued fifo-by-priority.

## 7.6. Continuous Simulations

7.6.1. "takes" ( exp action, time): action; #. duration specification

Takes specifies the approximate amount of time required to perform the action which is its first argument. Takes does not specify how the time is to be distributed over the takes interval. The implementation may distribute the time in any way it chooses consistent with wait and takes operations within the exp.

7.6.2. continuous( dim: dimension): type; #. continuous

Continuous may be used as the type of an attribute to indicate that the value of the attribute changes continuously over a takes interval. Assignments to a continuous attribute outside a takes, fix the value until the next assignment. References to a continuous attribute return the value at the current or a specified time.

7.6.3. "ref\_cont" ( x: exp continuous): x.dim; # ref continuous

When applied to a continuous attribute, the reference operation returns the value of the attribute at the time of the reference based on the assumption that the value changes linearly over the current takes interval including the time of reference.

7.6.4. "asgn\_cont" ( x: exp continuous, val: x.dim); # assign continuous

When applied to a continuous attribute, the assignment operation causes the value of the attribute to change linearly from its current value at the current time to the specified value at the currently enclosing takes time. If the assignment is outside a takes specification, The attributes value changes immediately to the specified value and remains there until the next assignment.

7.6.4. "ref" ( x: exp continuous, t: time): x.dim; #. ref linear interpolation

The two argument operation is used to reference the value of a continuous attribute at a specified time. The value is determined by linear interpolation or extrapolation from the last two assignments to the attribute.

[7.5.9.] "for" ( time, exp action, exp action): action; #. wait for time out

The time out operation specifies that execution of an action is to be terminated after a specified time interval whether or not it is complete. The interval is the first argument and the action is the second argument. If the interval is terminated by the time out the third parameter will be evaluated. Typically, the second argument would either directly or indirectly contain a wait condition and would time out if the condition does not become true within the specified interval.

[7.5.10.] continuous( time, exp action): action; #. continuous change

The continuous change operation specifies that an action is to be repeatedly evaluated for a specified duration and that the evaluation is to be in parallel with continued execution of the current actor. The action must include take or wait operations if within a simulation. The continuous operation allows unblocked operations without creating an additional actor.

## 7.7. Observers and Facilitators

7.7.1. observer: actor type; #. observer type



Observers are special actors whose activities are not bound by private and neighbor specifications. An observer may be used to collect information, interrogate simulations, or save simulation states. An observer can reference any attribute and apply any function without regard to private and neighbor restrictions.

7.7.2. facilitator: actor type;

#. facilitator type

Facilitators are special actors whose activities are not bound by private and neighbor specifications. A facilitator may be used to initialize and control simulations. A facilitator has the capabilities of an observer but may also assign to any attribute without regard to private and neighbor restrictions. The body of each simulation is a facilitator with respect to that and all embedded simulations. The body of a program is a facilitator with respect to all simulations of that program.

7.7.3. wait( simulation): action;

#. wait until simulation done

Only observers can wait for simulations to complete. Wait simulation suspends the simulation until all actors of the simulation are either terminated or permanently blocked from further execution.

7.7.4. terminate( simulation): action;

#. terminate simulation

Only facilitators can terminate simulations. Terminate immediately terminates the simulation including execution of all its actors. At the end of the body of each simulation there is an implicit call on wait sim followed by an implicit call on terminate sim. An explicit call on terminate can be used to bypass the wait.

7.7.5. suspend( actor | simulation): action;

#. suspend execution

Only observers can suspend actors and simulations. Suspend immediately suspends the specified actor or simulation. Suspending an actor will prevent any further increments to the clock of its simulation and thus will in effect suspend the simulation as well.

7.7.6. resume( actor | simulation, enum(

#. resume execution

run, step, step\_in, step\_out, stop): action; #.

Only observers can resume actors and simulations. Resume continues the execution of a suspended actor or simulation. The second argument specifies the duration of the resumption. Run indicates continuous execution. Step indicates resume for one statement of the body of the current operation. Step\_in indicates resume for one statement of the body of the next operation that is applied. Step\_out indicates resume until completion of the current statement of the caller of the current operation. Stop indicates resume until a program stop is encountered.

7.7.7. set\_stop( actor | nil, expression action): action; #. set stop

Only observers can set stops. Set stop marks the statement of the given actor to have a stop immediately before its execution. When execution reaches that point the actor will be suspended. If no actor is specified, the stop is placed at that position in all actors.

[7.7.8.] Context Navigation Operations 

scope: mutable type;

# visibility scopes

Scope is the supertype of all visibility scopes.

local( scope): scope;

# most local scope of actor

caller( scope): scope;  
eval( exp, scope): action | any type;

# caller's scope  
# eval exp in given scope

## [7.8.] Queuing.



### 7.8.2. semaphore: mutable type;

# semaphore

A semaphore is a low level mechanism for synchronizing among actors. Semaphores should be used only in situations in which the high level synchronizing mechanisms including neighbor relationships, conditional waits, timed waits and time outs are inadequate or when simulating existing computations that employ semaphores. Semaphores are mutable and thus must be created using the new operation. This semaphore mechanism includes at least four of the traditional forms of semaphores. Each semaphore has a count and a queue. The queue is a list of waiting actors. The count if positive is the number of actors that must be queued before any will be resumed, or if negative is the negative of the number of additional actors currently allowed in a critical region. The count field is initialized to negative one and the queue to an empty list.

### 7.8.3. enable( semaphore, n: int): action;

# enable semaphore

The enable semaphore operation sets the count field of the semaphore to n. Enable typically would be used only to initialize a semaphore that will be used to resume n actors simultaneously. The count will be restored to n each time the actors are resumed so there is no need to reenable the semaphore after each use.

### 7.8.4. psem( semaphore): action;

# p semaphore

The psem operation is a generalization of the traditional p operation on semaphores. If the count is less than zero, then the count is incremented by one. Otherwise, if the count is zero, the current actor is queued and suspended. Otherwise, if the count is one, then the count is set to the number of queued actors plus one, the queue is cleared, and all of the queued actors resumed. Otherwise, the count is decremented and the current actor is queued and suspended. Use psem on a non enabled semaphore to enter a critical region. Use psem on a negatively enabled semaphore to enter a critical region that allows -n actors simultaneously within the critical region. Use psem on a positively enabled semaphore when execution is to continue only when all n actors are waiting. Psem is an indivisible operation.

### 7.8.5. vsem( semaphore): action;

# v semaphore

The vsem operation is a generalization of the traditional v operation on semaphores. If the count is positive, then the count is incremented. Otherwise, if any actors are queued, the longest waiting actor is resumed. Otherwise, the count is decremented. Use vsem on a non enabled semaphore to exit a critical region. Use vsem on a negatively enabled semaphore to exit a critical region that allows -n actors simultaneously within the critical region. Use vsem on a positively enabled semaphore to increment the number of actors required for resumption. This latter use, allows the enable amount to be computed incrementally. Vsem is an indivisible operation.



## 8. Dynamic Graphic Depictions

Authors may define any number of views for an application or simulation. Actors may individually control their depiction within each view without central control. Users create windows and associate them with views. Depictions may be static drawing or dynamic drawing that vary throughout the course of a simulation. Drawings are often formed by painting a region with some pattern.

### 8.1. Views and Windows

8.1.1. `view( title: string, #. drawing view space  
background pattern, #.  
content list drawing): mutable type; #.`

A view is a perspective on a simulation. A view includes everything that can be seen from that perspective. Each view is an infinite (to the limits of the floating point representation) two-dimensional drawing space. Changes that occur to dynamic drawings during a simulated clock interval are not rendered in windows until completion of that interval. Each view belongs to exactly one simulation and is controlled by the clock of that simulation.

Each view has a title, a background pattern and a content list. The title is shown in the view menu when the program is run. Selection of an item in the view menu creates a window into the corresponding view. The background pattern fills the view in all regions that do not have drawings. The background may be specified as transparent to implement layers with different transformation properties. The content is a list of all drawings in the view in back to front order. Each view also has a private attributes indicating to which simulation the view belongs and the list of views referenced by projections and portals of the view.

### 8.1.2. Windows

Windows are generally in the province of users and are not manipulated directly by programs. Users may create any number of windows per view (including zero). Users control the size and location of the Window on the screen, and the position and magnification of the view within the window. Windows are created with the view origin in the upper left corner of the window (i.e., in the positive quadrant of the drawing space) and with magnification equal one.

## 8.2. Drawings

A drawing is any image that can be drawn in a view space. Drawings include painted regions, projections of views, portals to views, pictures, groups and text. Drawings may be read into the program at compile time using the include operation (see 1.1.6), read at run time using the read operation (see 9.1.4), or constructed within the program as described in this chapter. Drawings that are included or read may be constructed using the Easel graphic editor (see Appendix E) or by a foreign application.

8.2.1. `drawing: immutable type; #. the drawing type`

8.2.2. `paint( region, color | pattern): drawing; #. painted region`  
The paint operation creates a drawing by applying a color or pattern to a region.

8.2.3. `projection( region, view, view_transform): drawing; #. projection of a view`  
A projection is a portion of a view projected onto a region. The arguments to

projection are respectively, the region on which the view is projected, the view that is to be projected, and a transformation that is applied to the projected view to align it with the region. The region is specified in the coordinate system of the view that contains the resulting drawing. The transformation is applied to the projected view and then clipped to the region specified in the first argument to form the drawing. Projections are dynamic in that whenever the projected view changes, the projection changes accordingly. Transformations applied to a projection, apply to both the region and the projected content. Thus the displayed portion of the view remains constant relative to the region as might occur when a television set is turned upside down or movie projector is turned in a different direction.

8.2.4. `portal( region, view, view_transform): drawing;   #. portal into view`

A portal is similar to a projection except that transformations applied to a portal, apply only to the region and not to the view. Thus the displayed portion of the view changes relative to region as might occur when viewing the outside through the window of a moving car.

8.2.5. `picture( clip: region, array array color): drawing       #. picture`

A picture is a finite pixel drawing. The array specifies the color of all pixels of the enclosing rectangle of the clip region. Non rectangular regions may also be obtained by specifying some pixels as transparent. In any case, the picture is represented in standard GIF file format for input and output.

8.2.6. `depiction( v: view, d: exp drawing): mutable type;   #. type of depictions`

Depiction creates a new dynamic drawing and adds it to the specified view. The second argument is a expression that is reevaluated each time the value of any of its attributes change (but only at the end of each simulated clock interval). Subexpression of d that are not to be reevaluated may be marked with the value operation (see 2.1.2). Depictions have private attributes to maintain the current value of the depiction to keep track of which depictions must be recomputed and redrawn.

8.2.7. `update_depiction( depiction): action;               #. update depiction`

Calls on the `update_depiction` operation inform the system that the depiction must be recomputed at the next clock increment. `Update_depiction` is called automatically at least once in each clock interval in which any attribute of the depiction changes. Authors should never need to call `update_depiction` explicitly. NOTE however, in the initial implementation, `update_depiction` is not called automatically and thus authors must include at least one call per clock interval on each depiction for which some attribute change during that interval.

8.2.8. `transform( drawing | region,                       #. drawing transformations  
                  rotate: direction,                   #.  
                  scale: number,                       #.  
                  xoffset: distance, yoffset: distance, #.  
                  clip: region): drawing | region;   #.`

The drawing transformation operation rotates, scales, offsets and clips the first argument by the amounts given respectively to form the region that is returned. The transformations are applied in the order given. Argument values of zero, one, zero, zero, and everywhere respectively inhibit the corresponding portion of the transformation. Negative scales flip the region over both the x and y axes. This

operation may similarly be applied to regions to transform the region. NOTE: the initial implementation does not support transformations in or of projections or portals of views that contain text.

8.2.9. view\_transform( #. view transform  
rotate: direction, #.  
scale: number, #.  
xoffset: distance, yoffset: distance, #.  
clip: region): drawing; #.

A view transform is used in projections and portals to specify how drawing in the view is to be transformed through the projection or portal. Transformations may be applied to view\_transforms in the same manner as they apply to any other drawing.

[8.2.10.] flip( drawing | region, #. flip drawing or region  
slope: number, #.  
yintersect: number): drawing | number; #.

This operation flips a drawing or region over the line specified by slope and y intersect.

[8.2.11.] unscaled( drawing): drawing; # unscaled drawing

It is sometimes useful to have some drawings within a view that do not scale when the view is scaled. The unscaled drawing operation returns the unscaled version of its argument.

[8.2.12.] shared\_drawing( d: drawing): drawing; # shared dynamic drawing  
change\_drawing( shared\_drawing): drawing; # change shared drawing

A shared dynamic drawing preserves its identifier even when transformed. The may be any number of transformations of a shared\_drawing throughout the system. Shared drawings also may be dynamically changed by replacing their drawing argument. Such changes are reflected in all transformations of the shared\_drawing at the end of the current clock interval.

### 8.3. Groups

Groups are used to combine multiple drawings into a single drawing.

8.3.1. group( drawing... ): drawing; #. group drawings

The group operation creates a back to front list of drawings. Groups are actually drawn front to back in such a way that only uncovered pixels are drawn at each layer.

8.3.2. ungroup( drawing): list drawing; #. ungroup drawings  
Ungroup returns the drawings from which a group was formed.

8.3.3. bring\_to\_front( list, index: int): action; #. bring to front

This operation removes the indexed item of the list and appends it to the list. This operation may be applied to any list and is not restricted to lists of drawings.

8.3.4. send\_to\_back( list, index: int): action; #. send to back

This operation removes the indexed item of the list and inserts it at the beginning of the list. This operation may be applied to any list and is not restricted to lists of drawings.

[8.3.5.] bring\_to\_front( list, select: list int): action;      # bring to front  
 [8.3.6.] send\_to\_back( list, select list int): action;      # send to back

#### 8.4. Regions

A region is a portion of a view occupied by a drawing. Regions include polygons, lines and computed shapes.

8.4.1. region: immutable type;      #. region type

Each region is a portion of a view described to the precision of the floating point representation. Three operations are provided for creating regions depending on whether the region is a polygon, polyline or arbitrary shape.

8.4.2. empty: region;      #. empty region

Empty is a region constant indicating the region of zero size. The position of the empty region is arbitrary but by convention is always located at the origin.

8.4.3. everywhere: region;      #. universal region

Everywhere is a region constant indicating the infinite region that includes all points.

8.4.4. polygon( distance... ): region;      #. polygon

A polygon is a region defined by a sequence of three or more points. The points form the vertices of the polygon in the order given (the first point need not be repeated). Two arguments are required for each point (giving the x and y coordinates of the point respectively). When traversing the edges of the polygon in the order of the given points, points to the right of the edge are within the polygon; those to the left are outside. That is, the points describe the polygon in clockwise order. Edges of a polygon cannot cross each other.

8.4.5. polyline( w: distance, distance... ): region;      #. polyline

A polyline is a region defined by a width, w, and a sequence of two or more points. Two arguments are required for each point ( giving the x and y coordinates of the point respectively). Line segments connect each consecutive pair of points to form the polyline. The last point is not connected to the first. The region of a polyline extends w/2 distance to the right and left of each polyline segment plus a circle of diameter w around each point.

#### 8.4.6. Functional Shapes

A shape region is any function that returns a boolean given the x and y coordinates of a point. If true is returned the point is in the region, if false it is not. Shape functions can be used to define arbitrarily complex regions. Examples are:

```
positive_quadrant( x: distance, y: distance): boolean is      # positive quadrant
    return x > 0 & y > 0;
circle( radius: distance): region is      #. circle
    lambda( [ x: distance, y: distance], boolean,      #.
        x^2 + y^2 <= radius^2);      #.
oval( x1: distance, y1: distance,      #. oval
    x2: distance, y2: distance,      #.
    d: distance): region is      #.
```

```

lambda( [ x: distance, y: distance], boolean,      #.
        (sqrt (x-x1)^2 +(y-y1)^2) +              #.
        (sqrt (x-x2)^2 +(y-y2)^2) <= d);         #.

```

## 8.5. Operations on Regions

In addition to the operations in this section the transform operation (see 8.2.8) may be used to rotate, scale, offset and clip regions. Region intersect is not provided as a separate operation because it is equivalent to clipping with the transform operation.

8.5.1. `point_in_region( x: distance, y: distance, region): boolean; #. point in region`

Given a point and a region, `point_in_region` returns true iff the point is in the region

8.5.2. `region_of( drawing): region; #. region of a drawing`

This operation returns the region of the given drawing. For atomic drawings it is the existing region. For groups the region is computed as the union of the regions of the drawings in the group.

8.5.3. `"+"( region, region): region; #. region union`

The region union operation returns the combined region of its arguments.

8.5.4. `"-"( region, region): region; #. region difference`

The region difference operation returns the portion of its first argument that is not within its second argument. The first argument is returned if they do not intersect.

8.5.5. `"-"( region): region; #. region complement`

The region complement operation returns everything outside the given region.

## 8.6. Patterns

Drawings may be formed by filling a region with a pattern. Patterns may be single colors or a repeated pattern of colors.

8.6.1. `color( red: 0.. 255, green: 0.. 255, #. colors  
              blue: 0.. 255): type; #.`

This operation forms a color from specification of the saturation for each of the primary light colors. In the implementation colors are encoded as 24-bit rgb-colors and converted to 48-bit rgb-colors on output. The conversion is done by multiplying each color value by 0101#16.

## 8.6.2. Color Constants

Several color constants are provided. Authors may define others as appropriate. Transparent and inverse are system defined colors not otherwise definable. Transparent indicates that the color of the underlying region can be seen. Inverse indicates that the inverse of the color of the underlying region can be seen.

```

transparent: color;          #. transparent
inverse: color;             #. inverse
cream:    color is color( FC#, F3#, D4#);    #. cream
pistachio: color is color( AF#, FF#, AF#);    #. pistachio

```

```

aliceblue: color is color(F0#, F8#, FF#);    #. aliceblue
antiquewhite: color is color(FA#, EB#, D7#);  #. antiquewhite

```

aqua: color is color(00#, FF#, FF#);	#. aqua
aquamarine: color is color(7F#, FF#, D4#);	#. aquamarine
azure: color is color(F0#, FF#, FF#);	#. azure
beige: color is color(F5#, F5#, DC#);	#. beige
bisque: color is color(FF#, E4#, C4#);	#. bisque
black: color is color(00#, 00#, 00#);	#. black
blanchedalmond: color is color(FF#, EB#, CD#);	#. blanchedalmond
blue: color is color(00#, 00#, FF#);	#. blue
blueviolet: color is color(8A#, 2B#, E2#);	#. blueviolet
brown: color is color(A5#, 2A#, 2A#);	#. brown
burlywood: color is color(DE#, B8#, 87#);	#. burlywood
cadetblue: color is color(5F#, 9E#, A0#);	#. cadetblue
chartreuse: color is color(7F#, FF#, 00#);	#. chartreuse
chocolate: color is color(D2#, 69#, 1E#);	#. chocolate
coral: color is color(FF#, 7F#, 50#);	#. coral
cornflowerblue: color is color(64#, 95#, ED#);	#. cornflowerblue
cornsilk: color is color(FF#, F8#, DC#);	#. cornsilk
crimson: color is color(DC#, 14#, 3C#);	#. crimson
cyan: color is color(00#, FF#, FF#);	#. cyan
darkblue: color is color(00#, 00#, 8B#);	#. darkblue
darkcyan: color is color(00#, 8B#, 8B#);	#. darkcyan
darkgoldenrod: color is color(B8#, 86#, 0B#);	#. darkgoldenrod
darkgray: color is color(A9#, A9#, A9#);	#. darkgray
darkgreen: color is color(00#, 64#, 00#);	#. darkgreen
darkgrey: color is color(A9#, A9#, A9#);	#. darkgrey
darkkhaki: color is color(BD#, B7#, 6B#);	#. darkkhaki
darkmagenta: color is color(8B#, 00#, 8B#);	#. darkmagenta
darkolivegreen: color is color(55#, 6B#, 2F#);	#. darkolivegreen
darkorange: color is color(FF#, 8C#, 00#);	#. darkorange
darkorchid: color is color(99#, 32#, CC#);	#. darkorchid
darkred: color is color(8B#, 00#, 00#);	#. darkred
darksalmon: color is color(E9#, 96#, 7A#);	#. darksalmon
darkseagreen: color is color(8F#, BC#, 8F#);	#. darkseagreen
darkslateblue: color is color(48#, 3D#, 8B#);	#. darkslateblue
darkslategray: color is color(2F#, 4F#, 4F#);	#. darkslategray
darkslategrey: color is color(2F#, 4F#, 4F#);	#. darkslategrey
darkturquoise: color is color(00#, CE#, D1#);	#. darkturquoise
darkviolet: color is color(94#, 00#, D3#);	#. darkviolet
deeppink: color is color(FF#, 14#, 93#);	#. deeppink
deepskyblue: color is color(00#, BF#, FF#);	#. deepskyblue
dimgray: color is color(69#, 69#, 69#);	#. dimgray
dimgrey: color is color(69#, 69#, 69#);	#. dimgrey
dodgerblue: color is color(1E#, 90#, FF#);	#. dodgerblue
firebrick: color is color(B2#, 22#, 22#);	#. firebrick
floralwhite: color is color(FF#, FA#, F0#);	#. floralwhite
forestgreen: color is color(22#, 8B#, 22#);	#. forestgreen
fuchsia: color is color(FF#, 00#, FF#);	#. fuchsia
gainsboro: color is color(DC#, DC#, DC#);	#. gainsboro
ghostwhite: color is color(F8#, F8#, FF#);	#. ghostwhite
gold: color is color(FF#, D7#, 00#);	#. gold

goldenrod: color is color(DA#, A5#, 20#); #. goldenrod  
gray: color is color(80#, 80#, 80#); #. gray  
grey: color is color(80#, 80#, 80#); #. grey  
green: color is color(00#, 80#, 00#); #. green  
greenyellow: color is color(AD#, FF#, 2F#); #. greenyellow  
honeydew: color is color(F0#, FF#, F0#); #. honeydew  
hotpink: color is color(FF#, 69#, B4#); #. hotpink  
indianred: color is color(CD#, 5C#, 5C#); #. indianred  
indigo: color is color(4B#, 00#, 82#); #. indigo  
ivory: color is color(FF#, FF#, F0#); #. ivory  
khaki: color is color(F0#, E6#, 8C#); #. khaki  
lavender: color is color(E6#, E6#, FA#); #. lavender  
lavenderblush: color is color(FF#, F0#, F5#); #. lavenderblush  
lawngreen: color is color(7C#, FC#, 00#); #. lawngreen  
lemonchiffon: color is color(FF#, FA#, CD#); #. lemonchiffon  
lightblue: color is color(AD#, D8#, E6#); #. lightblue  
lightcoral: color is color(F0#, 80#, 80#); #. lightcoral  
lightcyan: color is color(E0#, FF#, FF#); #. lightcyan  
lightgoldenrodyellow: color is color(FA#, FA#, D2#); #. lightgoldenrodyellow  
lightgray: color is color(D3#, D3#, D3#); #. lightgray  
lightgreen: color is color(90#, EE#, 90#); #. lightgreen  
lightgrey: color is color(D3#, D3#, D3#); #. lightgrey  
lightpink: color is color(FF#, B6#, C1#); #. lightpink  
lightsalmon: color is color(FF#, A0#, 7A#); #. lightsalmon  
lightseagreen: color is color(20#, B2#, AA#); #. lightseagreen  
lightskyblue: color is color(87#, CE#, FA#); #. lightskyblue  
lightslategray: color is color(77#, 88#, 99#); #. lightslategray  
lightslategrey: color is color(77#, 88#, 99#); #. lightslategrey  
lightsteelblue: color is color(B0#, C4#, DE#); #. lightsteelblue  
lightyellow: color is color(FF#, FF#, E0#); #. lightyellow  
lime: color is color(00#, FF#, 00#); #. lime  
limegreen: color is color(32#, CD#, 32#); #. limegreen  
linen: color is color(FA#, F0#, E6#); #. linen  
magenta: color is color(FF#, 00#, FF#); #. magenta  
maroon: color is color(80#, 00#, 00#); #. maroon  
mediumaquamarine: color is color(66#, CD#, AA#); #. mediumaquamarine  
mediumblue: color is color(00#, 00#, CD#); #. mediumblue  
mediumorchid: color is color(BA#, 55#, D3#); #. mediumorchid  
mediumpurple: color is color(93#, 70#, DB#); #. mediumpurple  
mediumseagreen: color is color(3C#, B3#, 71#); #. mediumseagreen  
mediumslateblue: color is color(7B#, 68#, EE#); #. mediumslateblue  
mediumspringgreen: color is color(00#, FA#, 9A#); #. mediumspringgreen  
mediumturquoise: color is color(48#, D1#, CC#); #. mediumturquoise  
mediumvioletred: color is color(C7#, 15#, 85#); #. mediumvioletred  
midnightblue: color is color(19#, 19#, 70#); #. midnightblue  
mintcream: color is color(F5#, FF#, FA#); #. mintcream  
mistyrose: color is color(FF#, E4#, E1#); #. mistyrose  
moccasin: color is color(FF#, E4#, B5#); #. moccasin  
navajowhite: color is color(FF#, DE#, AD#); #. navajowhite  
navy: color is color(00#, 00#, 80#); #. navy



oldlace: color is color(FD#, F5#, E6#); #. oldlace  
 olive: color is color(80#, 80#, 00#); #. olive  
 olivedrab: color is color(6B#, 8E#, 23#); #. olivedrab  
 orange: color is color(FF#, A5#, 00#); #. orange  
 orangered: color is color(FF#, 45#, 00#); #. orangered  
 orchid: color is color(DA#, 70#, D6#); #. orchid  
 palegoldenrod: color is color(EE#, E8#, AA#); #. palegoldenrod  
 palegreen: color is color(98#, FB#, 98#); #. palegreen  
 paleturquoise: color is color(AF#, EE#, EE#); #. paleturquoise  
 palevioletred: color is color(DB#, 70#, 93#); #. palevioletred  
 papayawhip: color is color(FF#, EF#, D5#); #. papayawhip  
 peachpuff: color is color(FF#, DA#, B9#); #. peachpuff  
 peru: color is color(CD#, 85#, 3F#); #. peru  
 pink: color is color(FF#, C0#, CB#); #. pink  
 plum: color is color(DD#, A0#, DD#); #. plum  
 powderblue: color is color(B0#, E0#, E6#); #. powderblue  
 purple: color is color(80#, 00#, 80#); #. purple  
 red: color is color(FF#, 00#, 00#); #. red  
 rosybrown: color is color(BC#, 8F#, 8F#); #. rosybrown  
 royalblue: color is color(41#, 69#, E1#); #. royalblue  
 saddlebrown: color is color(8B#, 45#, 13#); #. saddlebrown  
 salmon: color is color(FA#, 80#, 72#); #. salmon  
 sandybrown: color is color(F4#, A4#, 60#); #. sandybrown  
 seagreen: color is color(2E#, 8B#, 57#); #. seagreen  
 seashell: color is color(FF#, F5#, EE#); #. seashell  
 sienna: color is color(A0#, 52#, 2D#); #. sienna  
 silver: color is color(C0#, C0#, C0#); #. silver  
 skyblue: color is color(87#, CE#, EB#); #. skyblue  
 slateblue: color is color(6A#, 5A#, CD#); #. slateblue  
 slategray: color is color(70#, 80#, 90#); #. slategray  
 slategrey: color is color(70#, 80#, 90#); #. slategrey  
 snow: color is color(FF#, FA#, FA#); #. snow  
 springgreen: color is color(00#, FF#, 7F#); #. springgreen  
 steelblue: color is color(46#, 82#, B4#); #. steelblue  
 tan: color is color(D2#, B4#, 8C#); #. tan  
 teal: color is color(00#, 80#, 80#); #. teal  
 thistle: color is color(D8#, BF#, D8#); #. thistle  
 tomato: color is color(FF#, 63#, 47#); #. tomato  
 turquoise: color is color(40#, E0#, D0#); #. turquoise  
 violet: color is color(EE#, 82#, EE#); #. violet  
 wheat: color is color(F5#, DE#, B3#); #. wheat  
 white: color is color(FF#, FF#, FF#); #. white  
 whitesmoke: color is color(F5#, F5#, F5#); #. whitesmoke  
 yellow: color is color(FF#, FF#, 00#); #. yellow  
 yellowgreen: color is color(9A#, CD#, 32#); #. yellowgreen

[8.6.3.] luminance( color, 0.. 100): color; #. luminance operation

The luminance returns a similar color to its parameter but in a darker or brighter form. A values in the range 75-100 are bright. Values in the range 35-50 are dark.



[8.6.4.] pattern: type is color | array array color; #. pattern

When a pattern is a two dimensional array of colors each array entry corresponds to a drawing pixel and the pattern is repeat in both dimensions as many times as necessary to fill the region. Position [0, 0] of the pattern is always aligned with the upper left corner of the screen when drawing.

## 8.7. Text

Text is a drawing that displays character data.

8.7.1. text( string, string\_class, list layout): type; #. text drawing

A text drawing consists of a string of characters, a string class, and as list of layout regions. The string is drawn within the layout regions according to the string class and the layout rules. Starting at the beginning of the string text is drawn in the first layout region. Text lines are folded at were boundaries to ensure that each line is no wider than the layout region. When no more lines will fit within a layout region, the text is continued at the next layout region. If there are no more layout regions, the remaining characters are not displayed. Text appears in drawing objects in the form of the layouts from which it is comprised.

8.7.2. layout( rectangle, background: pattern): drawing; #. text layout region

A text layout region consists of a rectangular region and a background pattern. Each layout also has private attributes specifying its associated text and starting index within the text's string. Transformation operations cannot be applied to drawings containing text, but may be applied to their enclosing view through a projection or portal. NOTE: the initial implementation does not support transparent text background.

8.7.3. rectangle( left: distance, top: distance, #. aligned rectangle  
right: distance, bottom: distance): type; #.

An aligned rectangle is defined by its left top point and its right bottom point. The resulting region is a rectangle with its sides parallel with the x and y axes.

8.7.3. string\_class: type is enum( simple\_text, html, xml); #. string class

The string class specifies the way in which the string will be interpreted. The simple text string class indicates that no formatting information is given in the string, that lines are to be drawn left justified, that carriage returns are to cause line folding, and that characters will be rendered in the application font and font size. Classes that support format information in the text allow specification and change of font, font size, font style, line height, edge aliment, text color and background pattern. Normally, the formatting specifications will be hierarchically structured with each embedded region will specify a text type that has semantic significance. The formatting information is then associated with the text type as a separate specification. Html and xml indicate that the string is to be interpreted as an html or xml specification respectively. NOTE: the initial implementation supports only simple\_text.

## 8.8. User Input

User input combines neighbor relations with graphics. The user can be thought of as an actor that can perform certain operations on the state of its neighbors. The operations however are not invoked by calls within the application, but by mouse and keyboard operations in displayed output. The user input operation provide a way of specifying which operations the user may perform, the associated graphic output, how

the operations are invoked and how the application will be informed.

8.8.1. `mouse_input( region | drawing, #. mouse input specification  
                  cursor | list cursor, #.  
                  click: lambda, drag: lambda): drawing; #.`

A mouse input drawing specifies the actions to be performed when the mouse is over the designated region or drawing in the front most window or in a dialog. It specifies the cursor to be displayed over the region and the neighbor actions to be executed in response to mouse clicks and drags. If the cursor, click action or drag action is unspecified, the mouse input specification has no affect on the response to that mouse actions. If the click or drag action is specified as none, there will be no response to the mouse action. Only one cursor, click action and drag action is defined for each region of the screen, with the front most specification hiding those behind it on a pixel by pixel basis. When the cursor is specified as a list, its display will cycle through the list changing images once per second.

8.8.2. `click_action( count: int, # response to mouse click  
                  x: distance, y: distance): action;`

Click\_action illustrates the signature for any operation passed as the click action to a mouse input specification. The author writes the operation which is called by the system when the mouse is clicked over the region of the mouse input specification and repeatedly for each release and subsequent multiple clicks. The count parameter specifies the number of clicks and releases with 1 being a click, 2 the release from that click, 3 a double click, 4 its release, etc. If the time between a mouse release and the next mouse click is greater than the double\_time system parameter of the Macintosh, the count is reset to 1. The point [x,y] is the point of the original click in the coordinate system of the view.

8.8.3. `drag_action( enum( begin, continue, end), # response to mouse drag  
                  prevx: distance, prevy: distance,  
                  x: distance, y: distance): action;`

Drag\_action illustrates the signature for any operation passed as the drag action to a mouse input specification. The author writes the operation which is called by the system when the mouse is dragged with its starting point over the region of the mouse input specification and repeatedly for subsequent moves until the mouse is released. The value of the first parameter will be begin on the initial call, continue on subsequent calls, and end on the final call. The point [x,y] is always the current mouse position in the coordinate system of the view. The point[ prevx, prevy] is the current mouse position on the previous call to drag\_action in this sequence. For the initial and final calls, the current and previous points are the same. Drag action continue to be reported for a drag even in the mouse position in no longer over the region of the mouse input region. In such cases the reported current position is on a nearby edge of the visible region.

8.8.4. `cursor( array( 16, array( 16, color)), x: int, y: int): type; #. cursor`

A cursor is a 16 by 16 array of pixels. Each pixel may be of any color including transparent and inverse. The pixel position [x,y] of the cursor is placed at the point of the mouse whenever the cursor is displayed. Exactly one cursor is displayed at all times.

8.8.5. `set_cursor( cursor): action;` `#. set cursor`

The set cursor operation may be called within a drag action to change the cursor while the mouse is down. When the mouse is released, the cursor image changes to that specified for the mouse release point.

8.8.6. `text_input( string, string_class, layout,  
start: int, stop: int,  
input_action: lambda ): drawing;`

Text input combines text output with keyboard input. As with `text_output` the given string is displayed in the layout region according the rules of the `string_class`. Text input also specifies that string positions start through stop-1 are to be selected by inverting their and their backgrounds color. If only a point is selected, it is indicated by a blinking vertical line insert point. When the layout is visible in the front window or in a dialog and is the last selected text input of that window or dialog, the user may select and edit its text using the mouse and keyboard input. When the user types a carriage return the input action is executed.

8.8.7. `input_action( ti: drawing): action;` `# text input action`

`Input_action` illustrates the signature passed as the `input_action` parameter to `text_input`. The author writes this operation which is called each time the user types a carriage return in the `text_input` layout region. The argument to input action routines is a modified text input drawing with the string, start and stop value updated to reflect the users edit actions.

## 9. Persistent Data

Persistent data is any data that resides beyond the execution of a program either in time or location. Persistent data includes data files, folders and print files.

### 9.1. Whole File Operations

A mutable file is any file that whose content can be changed. If a value returning operation fails for any reason not explicitly specified in its description, it will report the error and return nil. Failures within non value returning operations will generate an error report. No other error indication is given to the program.

9.1.1. `file( name: string, const st: storage, t: type): mutable type; #. file type`  
`file( t: type): type is file( ?, ?, t); #. file subtype`

Files are data that reside on peripheral storage. Each file has a string name, a storage unit on which it resides, and a content of type t. Files also have a parent file, universal id and a storage relative id as private attributes. References to files are valid whether the file is open or closed. References to existing files are obtained through the folder structure. Assignment to the name parameter of an immutable file (see section 5.4.2) changes its name. It is often convenient to categorize files by their content type.

9.1.2. `new( file type): file; # new file`

New creates a file of the specified type. The file type must be fully parameterized. This operation is a special case of the new operation on mutable types (see Section 5.4.3). The name of a file becomes the default name for the file in each folder that references the file. The resulting file will be open and mutable. Once a file is created it is allocated on the storage unit. The specified file name will appear in the underlying file system of the host computer only later when a reference to the new file is added to a folder. If a program creates a file but does not create a reference to the file in any folder, the file will be permanently inaccessible and thus will be removed from storage at program termination.

9.1.3. `make_file( folder | nil, name: string, t: type): file; #. make file`

Make file is a special operation that either creates a file or removes the content of an existing file. If the name references a mutable file of type t in the folder, the content of that file will be removed and the file returned as a closed mutable file. If the name does not reference any file in the folder, a file of the given name will be created, placed in the folder under that name, and returned as a closed mutable file. When a new file is created, its storage will be that of the folder and its type will be list. Make\_file is useful when creating a file that may or may not have previously existed, such as in repeatedly debugging or testing of programs. If the first parameter is nil, the folder will be root.application (see section 10.2.3).

9.1.4. `read( f: file): f.t; #. read entire file`

Read returns the entire content of file f. The file may be open or closed. This operation fails if the file has more than  $2^{15}$  records.

9.1.5. `write( f: file, value: f.t): action; #. write entire file`

Write replaces the entire content of the file by the value. The file must be mutable and the value must be an example of f.t. The file may be open or closed.

9.1.6. `open( file): boolean; #. open for exclusive write`

This operation opens a mutable file with exclusive write privileges. While the file is open the file cannot be written or changed by using any other reference to the file. Open returns true iff the operation is successful. Open need not be called for read or write operations where exclusive write privileges are not required. Open returns nil if the file is already open with exclusive write privileges. It reports an error if it is open with exclusive write privileges through this same file reference.

9.1.7. close( file): action; #. close exclusive write

Each call on close ensures that the actual file is closed at the operating system level. When close is paired with a corresponding open, close releases the exclusive write privilege. Any file that is opened for exclusive access but not closed will be automated closed at program termination.

## 9.2. Sequential File Operations

A sequential file is any file whose content type is sequence. Each element of the content sequence is a record of the file. At the implementation level easel supports only two kinds of sequential files, those containing sequences of characters and those containing sequences of any type. All foreign files are treated as character files. Sequential files may be incrementally modified only by appending new list items or removing some tail of the list.

9.2.1. reposition( file, increment: number): number; #. reposition file

Each sequential file has a read position that is set to 0 the first time the file reference is used in an application. The reposition operation increments the read position by its second parameter value. The parameter may be any value, but the resulting position will be constrained to the range 0.. length of file. Reposition returns the new value of the read position. Parameter values of -infinity, 0 and infinity respectively may be used to set to the beginning of the file, get the current read position and set to end of file.

9.2.2. read( f: file sequence, n: number): list f.t.t; #. read records

The read records operation reads the next n records from file f and increments the read position by the number of records read. For text files, each characters is a record. The records read are returned as a list. If there are less than n records remaining at the time of a read, all remaining records are read and returned. If an input-output error is encountered nil is returned. At most  $2^{15}-1$  records can be read at once.

Optimization: If read is called as an argument to apply (see sections 5.6.3 and 9.2.4), an optimization may combine the operations to eliminate the allocation of the read result.

9.2.3. read\_line( f: file string): string; #. read line

The read line operation reads the next line from file f and increments the read position by the number of characters read. The characters read are returned as a list. If an input-output error is encountered nil is returned. At most  $2^{15}-1$  characters can be read at once.

Optimization: If read\_line is called as an argument to apply (see sections 5.6.3 and 9.2.4), an optimization may combine the operations to eliminate the allocation of the read\_line result.

9.2.4. append( f: file list, sequence f.t.t...): action; #. append to file

Append inserts each item of each sequence at the end of the sequential file. The

sequence's items must be of the type required by the file. This operation is a generalization of the append list operation (see section 5.6.3). Append may be used to append several strings to a text file or printer output.

9.2.5. `write( f: file list, f.t.t...): action;` #. write records

Write adds each argument after the first as an additional record at the end of the sequential file. This operation is analogous to the push operation on lists (see section 5.6.4).

9.2.6. `truncate( file list, n: number): action;` #. truncate file

The file truncation operation removes all records of the file including and beyond record number *n*. The resulting content will be a list of *n* items. This operation is analogous to the truncate operation on lists (see Section 5.6.5). Truncate does nothing if the end of file is less than *n*.

### 9.3. Formatting and Printing

9.3.1. `format( file, any type... ): action;` #. format anything

The first argument to `format` is a destination file or list of characters. The remaining arguments are values of any type including expressions. `Format` returns the textual image of a program that would translate to the sequence of values. `Format` will also return the image of individual tokens and literals. For an arbitrary data structure, it will return the type qualified aggregate form that would generate the structure.

9.3.2. `format( int, file, any type...): action;` #. format structure

This form of `format` takes a extra first argument specifying special formatting rules. These include formatting as a structure regardless of the type. `Format` can also be used to print internal data structures that are generally inaccessible to application programs.

9.3.3. `output( any type...): action;` #. output text

`Output` converts each of its arguments in order from left to right to a string image and appends the result to the standard output file. `Output` is a special operation for conveniently outputting to the standard output file. `Output` works by calling `format` (see section 9.3.1) on `std_out` and each of its parameters.

9.3.4. `report( any type...): action;` #. report an error

The error reporting operation may be called by the system or by programs to report errors. Each of its arguments in order from left to right are converted to a string image and appended to the standard error file. `Report` works by calling `format` on `std_err` and each of its parameters.

### 9.4. Folders

Folders are a special form of file that provide a directory structure for referencing files. A file may appear in any number of folders on any number of storage units. The files of a folder need not be on the same storage as the folder. The name used to reference a file within a folder need not be the same as the files name.

9.4.1. `folder( storage, name: string): file type;` #. folder type

A folder is a file that contains references to files. More specifically, the content of a folder is a sequence of pairs. Each pair consists of string name and a file reference. The



same name cannot appear in two different pairs. Folders are created using the new operation.

9.4.2. `"."( folder, str_lit): file;` #. static folder ref  
`"dot"( folder, string): file;` #. dynamic folder ref

Dot qualification and quantification with a string arguments may be used to reference a file within a folder. If the folder does not contain a file paired with that name, nil is returned. The string name may be literal or computed.

9.4.3. `"."( folder, string, file): action;` #. static folder asgn  
`"dot"( folder, string, file): action;` #. dynamic folder asgn

Dot qualified assignment with a string argument is used to modify the content of a folder. The assign folder operation adds the string and file reference as a pair to the content of the folder. If a pair with the string name already exists in the folder it is removed. If the file parameter is nil, any previous pair by the string name is removed, but the new pair is not added. Adding or removing a file to a folder does not affect its storage unit. If however the system is able to determine that a reference to a file exists neither in memory nor in any folder, the file will be removed from storage.

9.4.4. `names( folder): list string;` #. folder names

The names operation returns a list of all of the string names that are paired with files in the folder.

#### [9.5.] Persistent Storage

Persistent storage is any memory device whose content persists beyond the execution of the program that created the content. Storage include hard disks of the host computer, removable storage media and central server storage. Printers and other output devices share many of the properties of storage and are thus treated as pseudo storage devices.

[9.5.1.] `storage( name: string): mutable type;` #. the storage type

Storage is the media on which files reside. Each storage has a string name. Each storage also has a universal id and a storage relative id as private attributes. The storage of a file or folder must be specified when it is created. Thereafter the storage only affects performance. The storage for a folder need not have any particular relation to the storage of files within the folder even if some or all are on removable storage. Storage is a pseudo file type that may be referenced from and only from the folder named "storage" within the system folder. The "storage" folder contains all currently active storage units.

[9.5.2.] `move( f: file, st: storage): file;` #. move file

The move operation moves the file to the specified storage and removes it from the original storage. Move may be applied to any file or folder whether mutable or immutable. The only attributes of the file changed by move are the storage and the storage relative id. Move is most often applied to mutable files.

[9.5.3.] `store( f: immutable file, st: storage): file;` #. store file

The store operation moves the file to the specified storage without removing it from the original storage. Store may be applied only to immutable files. The resulting files on the two storage units are indistinguishable, they have the same universal id, and any

reference to one is a reference to the other. They differ only in their storage and storage relative id.

[9.5.4.] `remove( f: file, storage): file type;` `#. remove file`

The remove operation removes the file from the storage. All immutable ancestors of `f` along `f`'s parent chain without intervening mutable ancestors, will also be removed from storage.

[9.6.] Version, Configuration and Release Control

The version, configuration and release control operations make use of parent which is a private attribute of each file. All file creation operations, other than copy and preserve, create files with nil parents.

[9.6.1] `remember( f: mutable file): action;` `#. remember version`

Preserve is used to mark a recovery point when making changes to a copied version of a file. Preserve makes an inaccessible copy of `f` and sets `f`'s parent attribute to reference the copy. The restore operation may be used later to recover the preserved version.

[9.6.2.] `recall ( f: mutable file): action;` `#. recall version`

Restore is used to recover to the most recently preserved but unrecovered file content. Preserve replaces the content of `f` by the content of `f`'s parent. If `f`'s parent is immutable, `f`'s parent is set to its grandparent, and the parent is removed from storage. Restore will fail if `f` does not have a parent. Removing a file with a preserved content will remove all preserved contents as well.

[9.6.3.] `copy( f: file, name: string, st: storage): file;` `#. copy file`

The copy operation creates a new version of the file. The copy has the same content and content type as `f`. The copy will be closed and mutable. The name of the copy will be the name parameter or the name of `f` if the name parameter is nil. The storage of the copy will be the `st` parameter or the storage of `f` if the `st` parameter is nil. The parent of the copy will be `f`. After all changes have been made to the new version's content, the new version may be frozen by a call on `make_immutable`.

[9.6.4.] `release( f: file): action;` `#. release new version`

This operation is used to freeze a version of a file. It makes the file permanently immutable for all users. If the file's parent is non nil, `make_immutable` also sets the file's parent to `f`'s youngest mutable ancestor and removes all younger immutable ancestors from storage. Note: ancestor files that are removed do not have universal ids and are inaccessible from applications.

[9.6.5.] `compare( file, file): action;` `#. compare and merge versions`  
`-- involves user interaction`



[9.6.6.] `merge( file, folder): action;` `#. compare and merge three way`  
`-- involves user interaction`

[9.6.7.] `check_out( file, list file): boolean;` `#. check out version`  
`-- combines copy with add file to list`  
`-- false says already checked out, i.e. on list`

[9.6.8.] `check_in( file, folder, list file): file | nil;` `#. check in version`  
`-- combines release with verify and remove file from list`





## 10. System Level Features

### 10.1. System Attributes

System attributes are shared among all simulations of a program. They include the random number seed, standard output files, and several real time attributes. There are also private system attributes containing the list of windows, the list of views, the list of enumeration types, and the minimum window update time interval.

#### 10.1.1. pronoun system: mutable is new { #. system attributes

System is a pronoun and thus may be referenced from anywhere in a program. The following system attributes may be referenced or assigned by dot qualification on system.

#### 10.1.2. seed: number := ?; #. random seed

The value of the seed attribute is the current seed for random number calculation within the program. The seed is automatically initialized to a different value each time the program is launched. The value of seed may be referenced and assigned by the program. Repeatable results can be guaranteed by assigning the same value at each execution.

#### 10.1.3. std\_out: file := make\_file( #. standard output file " ", "Console", string); #.

The value of std\_out is the file to which standard text output is written. Std\_out is initialized to the console file. The console window is created automatically the first time the console file is used and thereafter displays the content of the console file. The value of std\_out may be referenced and assigned by the program.

#### 10.1.4. std\_err: file := make\_file( #. standard error file " ", "Error Reports", string); #.

The value of std\_err is the file to which error reports are written. Std\_err is initialized to the error reports file. The error report window is created automatically the first time the error reports file is used and thereafter displays the content of the error report file. All errors reported by translator and interpreter are written to std\_err. Programs can report errors the same way. The value of std\_err attribute may be referenced and assigned by the program.

#### 10.1.5. assumptions: file := make\_file( #. interpreter assumptions " ", "Interpreter Assumptions", string); #.

The value of the assumptions attribute is the file to which interpreter assumptions are written. Assumptions is initialized to the interpreter assumptions file. The value of assumptions may be referenced and assigned by the program. The interpreter uses the assumptions file to report any assumptions it found necessary to make in order to continue execution. The assumptions can be checked by the author later and it desired make explicit in the program or explicitly denied in the program.

#### 10.1.6. std\_printer: storage := ?; #. standard printer

The standard printer is the application's currently selected printer. All currently

available printers may be referenced from the list named root.printers.

10.1.7. rtc: time := ?;  $\beta$  #. real time clock

The value of the real time clock system attribute is the current time of the physical world in the universal time (UT) zone. The same operations apply to real time as to simulated time (see Section 7.5).

10.1.8. time\_zone: time := ?;  $\beta$  #. time zone

The value of the time zone system attribute is the difference between the current time zone and universal time. By convention all absolute times are assumed to be ut unless converted to another time zone. The value of time\_zone is computed from the appropriate system parameter of the underlying operating system.

10.1.9. tick\_clock: number := ?;  $\beta$  #. tick clock time

The value of the tick clock is the number of sixtieth of a seconds since the current program was activated. The tick clock cycles approximately once every 77.67 hours (i.e., every  $2^{23} 1/60$ th's of a second).

10.1.10. microsec\_clock: number := ?;  $\beta$  #. microsecond clock

The microsecond clock increments once per microsecond. It is very precise, but cycles approximately once every 16.78 seconds (i.e., every  $2^{24}$  micro seconds).

10.1.11. update\_interval: time := (1/60) \*second; }; #. window update interval

The update interval is the minimum real time between consecutive updates of a window.

10.2. System Root  $\beta$

The system root is a means for programs to access the persistent data files, storage volumes, printers and other devices of the underlying computer system.

10.2.1. pronoun root: mutable is new { #. system root

The system root is a pronoun that serves as the root of the current user's persistent data space. All data files, storage volumes and peripheral devices accessed by the program must be directly or indirectly accessible through root.

10.2.2. user: folder := ?; #. user folder

The user folder is the root of all files accessible by the current user.

10.2.3. application: folder := ?; #. application folder

The application folder is the working (i.e., unprefixd) folder for the current execution. It is initialized to the folder from which the application was launched, either the folder containing the application itself (when launched by double clicking or menu selection of the application) or the folder containing the application data (when launched by double clicking or menu selection of the data file).

10.2.4. volumes: list storage := ?; #. storage volumes

The value of the volumes attribute is a list of the current storage devices that are accessible on the underlying physical computer system. Volumes is update each time it is referenced.

10.2.5. printers: list storage := ?; }; #. printers

The value of printers is a list of all printers currently accessible to the program. A printer is a pseudo storage device referencing a physical printer. Duplicating a file on a printer or moving a file to a printer causes the file to be printed and then removed from the printer storage. For semantic consistency an immutable file also may be copied to a printer. Files also may be created on a printer storage using the new operation, in which case the file will be printed and removed from the printer storage when the file is closed. The printers attribute is updated each time it is referenced.

### 10.3. Translator

- lexical analysis
- parsing
- semantic analysis
- code generation

### 10.4. Interpreter

- application loop
- tail recursion
- overload resolution
- error propagation
- symbolic computation

### 10.5. Memory Management

- memory allocation
- pwd( int, int, int): any type;
- garbage collection

### 10.6. Windows

Windows are updated at most once every sixtieth of a second. Only those portions of a window that has been uncovered or whose content has changed are redrawn. All drawing is front to back with each pixel being drawn only once. All drawing is done with pixel graphic regions for high performance.

10.6.1. window: drawing type; # the window type

Although windows could be implemented as projection drawings of the screen, for application level compatibility they are instead implemented as windows of the underlying operating system. Each window is updated whenever portions of it are uncovered or its content changes, but never more frequently than the update interval. Content changes are displayed only at the end of clock intervals of the associated simulation. Only regions of the window that have been changed or uncovered are redrawn.

10.6.2. "draw"( window): action; # screen update operation

Draw is an internal operation that is called by the run time system and cannot be explicitly called by application programs. Draw is called by the event manager when a portion of a window is uncovered. It is also called by the scheduler whenever the simulated clock is incremented and some view of that simulation has changed. Draw is a recursive function that scans the content of the window starting from the view. It applies the specified transformations at each level of the recursion. Processing is optimized by drawing front to back only those components of the view that have a non

empty intersection with the update region, by drawing only those portions that intersect with the update region, and by reducing the update region as portions are drawn.

10.6.3. "invalidate" (simulation): action; # invalidate window regions

Invalidate is an internal operation that is called by the interpreter and cannot be explicitly called by application programs. Invalidate is called by the scheduler whenever the simulated clock is updated and some view of the simulation has changed. Changes in views are determined by a non empty list of changed depictions in the simulation at the time of clock increment. Invalidate checks each window that contains projections or portals to views of the simulation. It computes and invalidates any region of the visible portion of the window that displays a depiction whose image has changed since the previous simulated clock increment. It then copies any new images of depiction to the old image attribute of the depiction. If however invalidate is called less than an update interval from the previous update of a window, the change region is not invalidated, and instead the new depiction images are computed and assigned as the new (but not old) images. The effect is that both the invalidation and drawing processes are skipped when the real time of a simulated clock interval is less than the update interval.

]10.7.] Event Handling 

[10.8.] Foreign Code Interface

The foreign code interface provides a means for importing object code for operations implemented in languages other than Easel.

[10.8.1.] foreign( string): any type; # include foreign code

The include foreign code operation is executed at compile time. Foreign may be called only as the body of the definition of the operation. Its argument must be a string literal. The string must be either an absolute or relative file path within the folder structure. The implied prefix for relative paths is the folder from which the reference to the file containing the include was obtained. The resulting file must be an object code file containing an implementation of the operation being defined.

[11.] Example Programs 

11.1. positional

11.2. mobile

11.3. thing

11.4. gcd( int24, int24): number;

11.5. tree\_sort3

11.6. alpha\_beta


# all physical things

# greatest common divisor

# ln n priority queueing

# alpha beta search

A. Syntax and Grammar

A.1. Lexical Grammar 

The lexical grammar specifies the sequences of source program characters that constitute words of the language. The lexical grammar is formally defined using the meta grammar of section A.4.

A.1.1. Identifiers

id ::= alpha [{ ["\_"] alphanumeric }]

alpha ::= "a" | "b" | ... "z" | "A" | "B" ... "Z"

alphanumeric ::= alpha | "0" | "2" | ... "9"

Identifiers are used as names to reference values within a program. Easel does not distinguish between upper case and lower case alphabetic characters in identifiers.

#### A.1.2. Operators

op ::= "

#### A.1.3. Numeric Literals

int\_lit ::= { digit }

# integer literal

num\_lit ::=

# numeric literal

[[{int\_lit}] "#"]

# base

["~"] { digit }

# integer part

["."] { digit }

# fractional part

[("e" | "E") ( "+" | "-" ) int\_lit ]

# exponent part

An integer literal is a sequence of one or more decimal digits interpreted as a decimal number. A numeric literal may be signed, in any base 1.. 36, have a fractional part and an optionally signed exponent. The base specifies the radix for interpreting the integer and fractional parts. Decimal may be specified by omitting the radix specification and base 16 by including only the "#". The tilde as a sign is a special feature for use in data; in programs the negate operator is more readable. The integer and fraction parts are interpreted together as a number of the specified radix. Each digit must be less than the radix where the value of decimal digits in 0.. 9 respectively, and of alphabetic characters is 9 plus their ordinal position in the alphabet. The exponent is specified as a possibly signed decimal number. It is interpreted as a exponent of the base. That is the preceding number is multiplied by the base to the power of the exponent. If either the fraction or exponent are omitted, their value is taken as zero.

#### A.1.4. String Literals

str\_lit ::= "\q" [{ printable\_char }] "\q"; # string literals

printable\_char ::= " " | "!" | "#" .. "[" | "]" .. "~"

| "\" ( "r" | "t" | "q" | "\" | digit digit

String literals have a special lexical form (see section 5.10.0). A string literal is any sequence of printable ascii characters enclosed in by double quotes, ", with \ and " omitted. A \ may appear in a string literal as a control character. Followed by r, t, q or \, the pair stands for carriage return, horizontal tab, double quote mark and back slant respectively. Followed by two hex digits, the triplet stands for the ascii character of that value.

#### A.1.5. Comments

#### A.1.6. Fill Space

#### A.1.7. Key Words

#### A.1.8. Indenting Rules

Each line of a program begins with some sequence of tab and space characters. The amount of white space at the beginning amount determines the number of embedded scope levels of that line within the program. The absolute amount in irrelevant, but the relative amounts of which lines are more embedded. All lines in the same scope should have exactly the same prefix of tabs and spaces. In addition, "{" "}" pairs may be used to surround program scopes. Either or both methods may be used for a given scope.

More precisely, in computing the indent amount the lexical analyzer assumes that tabs have much more white space than spaces, it ignores any spaces preceding a tab,

and the amount of leading white space on continuation lines (see section A.1.9) does not matter.

### A.1.9. Line Continuation

Any logical line may be folded into multiple physical lines. All but the first physical line of a logical line are called continuation lines. The amount of leading white space on continuation lines does not affect the indenting scope of the logical line. No markers are required to indicate continuation lines in Easel; a carriage return is adequate.

Formally, a continuation line is any line following a line that does not end in one of the following eight words: `•••`, or that begins with `"}` or `"{`. In determining what begins or ends a line, tabs, spaces, fill characters and comments are ignored.

### A.2. Semantic Grammar

```

prog ::= { prop | stat } [ exp]           # easel program
prop ::= "property" exp                   # predicate or supertype property
      | id ":" type [ "is" body ]         # definition
      | name [ fpl ] ":" type [ "is" body ] # lambda definition
      | name [ fpl ] "is" body            # definition with unspecified type
      | id ":" type ":" exp               # attribute definition
      | exp "can" body                    # neighbor relationship
      | id prop                           # computed property
      | stat                              # statement
      | exp "." exp ":" type ":" exp      # indexable attribute definition
      | "attribute" type "." id ":" exp   # external attribute definition
type ::= exp                             # type valued expression
body ::= "{" [ { prop "," } ] "}"         # body of type definition
      | exp | prop                        # simple expression, property or statement as body
name ::= id | "\"q\" op \"q\"             # name: id or quoted operator
      fpl ::= "(" [ { fp "," } ] fp [ "..."] ")" # formal parameter list
      fp ::= [ id ":" ] type              # formal parameter

exp ::= id | num_lit | str_lit            # atomic expressions
      | "(" exp ")"                       # parenthesized expression
      | exp op exp | op exp | exp "..." # in-fix, prefix, and postfix expression
exp ::= exp "(" ")"                       # lambda application - zero args
      | exp exp                           # lambda application - one arg
      | exp "(" exp { "," exp } ")"       # lambda application - multiple args
      | quantifier [ { exp } ] type        # quantification
      | exp "." id | exp "." int_lit      # dot qualified reference
      | exp "." "(" exp ")"               # dot quantified reference
      | exp "." "[" exp "]"               # indexed reference
      | "[" [ { exp "," } ] exp "]"        # aggregate constructor
      | "for" [ id ":" ] exp "do" exp     # quantified condition
      | "?"                               # unspecified value

pronoun ::= "it" | "this" | "self" | "sim" | "system" # pronouns
op ::= "<<" | ">>" | "&" | "|" | "!" | "isa" | "~" # type operators
      | "<" | "=" | ">" | "<=" | "!=" | ">=" | ".." # relational operators

```

```

| "+" | "-" | "*" | "/" | "^"      # computational operators
| "-" | "$" | "%" | "\"          # unassigned operators
| "can"                          # special syntax operations
quantifier ::= "any" | "some" | "the" | "all" | "each" | "every" | num_lit

stat ::= "?"                      # unspecified action
| "{" [{ prop ";" } ] "}"        # compound statement
| exp [ "(" ")" ]                # procedure call - no args
| exp exp                        # procedure call - one arg
| exp "(" exp { "," exp } ")"    # procedure call - multiple args
| exp ":" exp                    # assignment statement
| "if" exp "then" stat [else_part] # conditional statement
| "for" [ id ":" ] exp "do" stat [else_part] # iterative statement
| "select" exp of [{ "case" type "then" stat } ] [else_part] # select statement
| "take" exp "to" stat           # duration specification
else_part ::= ";" "else" stat

```

### A.3. Parse Grammar

### A.4. Meta Grammar

The following meta grammar is used to define both the lexical grammar and parse grammar.

abc	any alphabetic sequence name a lexical category
"xyz"	xyz is a terminal of the language being defined
x ::= y	y is a legal expansion of category x
x y	x juxtaposed with y in that order
x   y	x and y are alternative choices
[ x ]	x is optional
{ x }	x may be repeated one or more times
[{ x }	x may be repeated zero or more times
x1 ... xn	short hand for x1   x2   x3   x4   ... xn

## B. Messages and Reports



### B.1. Error Messages

```

-- any inconsistency # inconsistency
-- reference uninitialized attribute          # uninitialized attribute
-- cannot resolve      # unresolved ambiguity
-- computation errors normally return an error value # computational error

```

### B.2. Assumptions

## C. Running and Controlling Simulations



- general procedures
  - call program with actual parameters
  - may also call simulations without intervening program
- simulation controls
- command line interface
  - user has full facilitator capability on any simulation executed
  - nominally in context of built-in definition of language definition
  - as if were in outer level of a program



- each decl, stat, and exp evaluated as entered in this pseudo program context
- declarations can precede statements
- expressions show value in command line window

#### D. Debugger

- windows, menus and controls
- special commands only from command line for:
- single step simulation clock
- suspend execution
- move around in dynamic structure of program
- display attributes and values of a scope

#### E. Graphic Editor

##### E.1. Mouse Operations

Graphic editing may be done in any graphic edit window. The mouse operations of this section are used to create and transform drawings. Selected drawings are highlighted by drawing a square corner at each point that the drawing intersects its own enclosing rectangle.

Each mouse operation involves either a click or drag. They may also involve the use of modifier keys: shift, option, command or control. The modifier keys that initiate operations need be depressed only during the mouse down and apply throughout the operation. For certain operations shift and command have significance at the point of mouse up.

##### E.1.1. cmd mouse up # cancel mouse op

If the command key is depressed during mouse up of any mouse operation of Section E.1, the entire operation will be canceled and the drawing state returned to what it was before the corresponding mouse down.

##### E.1.2. cmd click # set anchor cmd drag # create drawing

This operation is used to set anchors and create drawings. At mouse down any previous anchor is removed and the anchor set and displayed at the mouse down point. At the first drag, all selected drawings are unhighlighted and deselected, and a new zero size drawing is created, selected, and highlighted at the anchor point. During the drag the selected drawing is resized to the distance between the anchor and current drag position. The shape of the drawing is the last selection made in the shape menu (see Section XXX). The paint is the last selection made in the paint menu (see Section XXX). At mouse up, if there has been a drag and the current point is at the anchor point, the drawing is unhighlighted, deselected and deleted. Otherwise the current state is preserved.

##### E.1.3. click on unselected drawing # select at point click on background # deselect all drag from background or unselected drawing # select in region

At mouse down, all previously selected drawings are unhighlighted and deselected, and a zero size select box is created and drawn at the point. During the drag the select box is resized and redrawn with opposite corners at the start and current mouse points, and all unselected drawings that intersect any point of the select box are highlighted.



At mouse up all unselected drawings that intersect the select box are selected and the select box is removed.

E.1.4. shift click on drawing	# toggle select
shift click on background	# deselect all
shift drag	# add to select

At mouse down, if there is a drawing at the point, the select state and highlighting of the top most drawing at the point is toggled. A zero size select box is created and drawn at the point. During the drag the select box is resized and redrawn with opposite corners at the start and current mouse points, and all unselected drawings that intersect any point of the select box are highlighted. At mouse up all unselected drawings that intersect the select box are selected and the select box is removed.

E.1.5. drag from interior of selected drawing	# move drawing
---	----------------

It moves the selected drawings by the distance and direction that the mouse moves from the start point to the current point. Mouse up preserves the new positions. This operation does not change which drawings are selected.

E.1.6. drag from corner of selected drawing	# resize drawing
---	------------------

It magnifies the selected drawings with the anchor as the fixed point and the magnification as the ratio of the distances from the current point and start point to the anchor. If shift is depressed during the drag and mouse up, separate calculations will be used for the x and y magnifications, allowing drawings to be stretched. Mouse up preserves the new size and current select state. This operation does not change which drawings are selected.

E.1.7. ctrl drag	# rotate drawing
------------------	------------------

Use this operation to rotate the selected drawings around the anchor. Nothing happens at mouse down. During the drag the selected drawings are rotated by angle between lines from the mouse down point and from the current drag point to the anchor. Mouse up terminates the operation.

E.1.8. option drag from corner of selected polygon	# move corner
--	---------------

This operation applies only if the top currently selected drawing is a polygon and the moused down position is near a corner of that polygon. At mouse down, all drawings are unhighlighted and deselected. During the drag, the corner is moved to the current drag position and the polygon is redrawn. At mouse up, if the corner is on a line connecting its adjacent corners, the corner is removed. If the polygon then has less than three sides, the polygon is deleted. Otherwise the polygon is selected and highlighted.

E.1.9. option drag from corner of selected polygon	# move new corner
--	-------------------

This operation applies only if the top currently selected drawing is a polygon and the moused down position is near an edge of that polygon. At mouse down, all drawings are unhighlighted and deselected, and a new corner is added to the polygon on the near edge, at the point of the mouse down. During the drag, the corner is moved to the current drag position and the polygon is redrawn. At mouse up, if the corner is on a line connecting its adjacent corners, the corner is removed. The polygon is selected and highlighted.

E.1.10. option drag from selected drawing # copy drawing



Nothing happens at mouse down. At the first drag, all selected drawings are copied, the originals are unhighlighted and deselected. The copies are moved to the front in the order of the originals, selected and highlighted. During the drag the new selected drawings are moved by the distance and direction from the mouse down point to the current position. Mouse up preserves the new selected copies at the new position.

E.1.11. option click on selected, all at back # bring to front  
option click on selected, all at front # send to back  
option click on selected, not all together # bring forward

This operation does not change which drawings are selected, the front to back order of the selected drawings, nor the front to back order of the unselected drawings. Nothing happens at mouse down. At the mouse up, if the selected drawings are not contiguous in their front to back order, they are brought forward until they are contiguous without changing the position of the top most selected drawing. Otherwise, if the top drawing is selected, all selected drawings are moved to the back. Otherwise, all selected drawings are moved to the front. In any case, the window is redrawn.

E.1.12. opt click on background, multiple drawings selected # group  
opt click on background, one group selected # ungroup  
opt click on background, one nongroup selected # complement

If multiple drawings are selected, they are unhighlighted, deselected, removed from the view list, and collected into a single group object in the same order. The resulting drawing is inserted into the view at the top position of the previous selection, the window is redrawn, and the group is selected and highlighted. If a single group drawing is selected, it is unhighlighted, deselected, removed from the view list, and separated into its component drawings. The component drawings are then inserted into the view at the previous point of the group and in the same front to back order, and are selected and highlighted. If a single non group drawing is selected, in is unhighlighted, deselected, replaced by its complement, the window redrawn, the complement selected highlighted. Otherwise this operation does nothing.

E.1.13. Line Drawing  

## E.2. Keyboard Commands

E.2.1. delete key with selection # delete selection  
delete key without selection # show/hide anchor

When the front window is a graphic editing window, this operation unhighlights, unselects and removes all selected drawings, and updates the affected windows. If no drawings are selected, delete toggles the show/hide state of the anchor. Delete does not have a menu or mouse equivalent.

E.3. Draw Menu  

The draw menu is activated whenever the front window is a graphic edit window. If no drawing within the window is selected, the draw menu gives the default settings that are used when a new drawing is created. Otherwise, the draw menu gives the current values for the selected object. The draw menu has three parts. The first part gives the current values for the type, shape, paint or view, and background. The

second part has graphic edit operations that are either unavailable or have more options than are available with the mouse alone. The third part has graphic editing and selection operations that are also available as mouse or keyboard operations.

#### E.3.1. Type: XXX...

Type is a submenu of Draw and is activated whenever Draw is activated. Type specifies the type of the current drawing. The following choices are available: Paint, Portal, Projection, Text, and Group. The current selection is displayed instead of the XXX above. The selected item is also checked in the submenu. Group is dimmed and not selectable as the choice when creating new drawings.

#### E.3.2. Shape: XXX YYY...

Shape is a submenu of Draw and is activated whenever Draw is activated. Shape specifies the shape of the current drawing. The choices include polygons, lines and arbitrary computed shapes, and are described in section E.4. The current selection is displayed instead of the XXX above. The selected item is also checked in the submenu. The YYY above is displayed as empty or "Outline" depending on whether the selection is fill or outline respectively.

#### E.3.3. Paint: XXX...

Paint is a submenu of Draw and is activated whenever the currently selected Type is Paint. Paint specifies the content of the region of the current drawing. The choices include

Paint...	New Color...
	New Pattern...
	Red, Green, Blue, Yellow, Magenta, Cyan, Black, White,
View...	New View...
Background...	Red, Green, Blue, Yellow, Magenta, Cyan, Black, White,
	New Color...
Clip	menu clip with two selected: clip bottom selected to region of top selected, delete top selected
Difference	menu diff with 2 selected: clip bottom selected to complement of top selected, delete top selected
Rotate...	
Scale...	
Nudge...	
Group	
Bring to Front	
Send to Back	
Show Anchor	
Duplicate	
Complement	
Delete	

#### E.4. The Shape Submenu

The shape submenu of Draw is described in section E.3.2. This section describes the items within the Shape submenu. The Shape menu is divided into XXX parts. The first

part is used the change or extend items within the menu and consists of the following items: New Polygon... and New Shape.... The second part specifies whether the shape is filled or outlined. The third part contains the available polygon choices and consists of the following items: Triangle, Quadralateral, Pentagon, Hexagon, Octagon, and XXX Point Polygon where XXX is the last value of New Polygon. The fourth part contains the available shapes other than polygons and consists of the following items: Line, Free Hand, Oval, Sine, Fractile, and other items as defined by New Shape.

E.4. xxx NOT YET DESCRIBED.

Windows

Center View...

Find Drawing...

E.4. Shape Menu



E.5. Paint Menu

set default paint      menu paint without selection:

menu paint with selection: compute union of select regions,  
paint region, replace top select, delete other selects

E.6. Text Menu



[F.] CGI Interface



Progress Markings:



Future features under consideration, will not be in beta release



Temporary version currently available, not as to spec



ELRM description under construction and is not reliable



Description unreliable, feature unlikely to be retained, at least in this form.